**MỤC LỤC**

[MỞ ĐẦU 5](#_Toc207267401)

[1. Lý do chọn đề tài 5](#_Toc207267402)

[2. Mục tiêu của đề tài 5](#_Toc207267403)

[3. Đối tượng và phạm vi nghiên cứu 6](#_Toc207267404)

[4. Bố cục của báo cáo 6](#_Toc207267405)

[CHƯƠNG 1: TỔNG QUAN TÌNH HÌNH NGHIÊN CỨU 8](#_Toc207267406)

[1.1. Bối cảnh phát triển Thương mại điện tử tại Việt Nam 8](#_Toc207267407)

[1.2. Xu hướng mua sắm thực phẩm và hàng tiêu dùng trực tuyến 8](#_Toc207267408)

[1.3. Các công nghệ phổ biến trong xây dựng Website TMĐT hiện đại 9](#_Toc207267409)

[CHƯƠNG 2: CƠ SỞ LÝ LUẬN 11](#_Toc207267410)

[2.1. Tổng quan về Thương mại điện tử (E-commerce) 11](#_Toc207267411)

[2.1.1. Khái niệm 11](#_Toc207267412)

[2.1.2. Các mô hình TMĐT phổ biến 11](#_Toc207267413)

[2.1.3. Các thành phần chính của một Website TMĐT 11](#_Toc207267414)

[2.2. Công nghệ sử dụng trong đồ án 12](#_Toc207267415)

[2.2.1. ReactJS - Thư viện xây dựng giao diện người dùng 12](#_Toc207267416)

[2.2.2. Google Firebase - Nền tảng Backend-as-a-Service (BaaS) 13](#_Toc207267417)

[CHƯƠNG 3: CƠ SỞ DỮ LIỆU 15](#_Toc207267418)

[3.1. Sơ đồ Quan hệ Thực thể (ERD) 15](#_Toc207267419)

[3.2. Mô tả chi tiết các Collection 19](#_Toc207267420)

[3.3 Sơ đồ lớp (Class Diagram) 23](#_Toc207267421)

[CHƯƠNG 4: PHÂN TÍCH VÀ XÂY DỰNG HỆ THỐNG 24](#_Toc207267422)

[4.1. Cấu trúc Tổng thể của Ứng dụng 24](#_Toc207267423)

[4.2 Sơ đồ Usecase tổng quát 25](#_Toc207267424)

[4.3 Sơ đồ hoạt động 27](#_Toc207267425)

[4.3.1 Sơ đồ hoạt động của khách hàng 27](#_Toc207267426)

[4.3.2 Sơ đồ hoạt động của quản trị viên/ nhân viên 37](#_Toc207267427)

[4.4 Sơ đồ tuần tự 47](#_Toc207267428)

[4.4.1 Sơ đồ tuần tự của khách hàng 47](#_Toc207267429)

[4.4.2 Sơ đồ tuần tự của quản trị viên 57](#_Toc207267430)

[4.5. Module Giao diện Khách hàng 65](#_Toc207267431)

[4.5.1. Trang chủ (HomePage) và Hiển thị Sản phẩm 65](#_Toc207267432)

[4.5.2. Giỏ hàng (CartPage) và Thanh toán (CheckoutPage) 66](#_Toc207267433)

[4.5.3. Tài khoản Người dùng (ProfilePage) 68](#_Toc207267434)

[4.5.4 Chi tiết sản phẩm (ProductDetailPage) 71](#_Toc207267435)

[4.6. Module Hệ thống Quản trị (Admin) 71](#_Toc207267436)

[4.6.1. Phân quyền và Layout 71](#_Toc207267437)

[4.6.2. Chức năng của Admin chính 71](#_Toc207267438)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN HỆ THỐNG 75](#_Toc207267439)

[5.1. Kết luận 75](#_Toc207267440)

[5.1.1. Những kết quả đạt được 75](#_Toc207267441)

[5.1.2. Hạn chế của hệ thống 75](#_Toc207267442)

[5.2. Hướng phát triển hệ thống 76](#_Toc207267443)

**DANH SÁCH BẢNG BIỂU**

[Bảng 3.1 Mô tả sơ đồ quan hệ thực thể (ERD) 16](#_Toc207267308)

[Bảng 4.1 Mô tả sơ đồ hoạt động: Đăng ký và đăng nhập 27](#_Toc207267309)

[Bảng 4.2 Mô tả sơ đồ hoạt động: Đổi điểm thưởng lấy Voucher 29](#_Toc207267310)

[Bảng 4.3 Mô tả sơ đồ hoạt động: Gửi đánh giá sản phẩm 31](#_Toc207267311)

[Bảng 4.4 Mô tả sơ đồ hoạt động: Mua hàng của khách hàng 34](#_Toc207267312)

[Bảng 4.5 Mô tả sơ đồ hoạt động: Quản lý danh sách yêu thích 35](#_Toc207267313)

[Bảng 4.6 Mô tả sơ đồ hoạt động: quản lý tài khoản cá nhân 36](#_Toc207267314)

[Bảng 4.7 Mô tả sơ đồ hoạt động: Nhập kho hàng loạt từ Excel 38](#_Toc207267315)

[Bảng 4.8 Mô tả sơ đồ hoạt động: Xử lý đơn hàng 39](#_Toc207267316)

[Bảng 4.9 Mô tả sơ đồ hoạt động: Quản lý & Phân quyền người dùng 41](#_Toc207267317)

[Bảng 4.10 Mô tả sơ đồ hoạt động: Quản lý nội dung (CMS) 42](#_Toc207267318)

[Bảng 4.11 Mô tả sơ đồ hoạt động: Quản lý sản phẩm & Gán nhãn hiệu hàng loạt 44](#_Toc207267319)

[Bảng 4.12 Mô tả sơ đồ hoạt động: Thêm sản phẩm mới 47](#_Toc207267320)

[Bảng 4.13 Mô tả sơ đồ tuần tự: Cập nhật thông cá nhân 48](#_Toc207267321)

[Bảng 4.14 Mô tả sơ đồ tuần tự: Đăng nhập người dùng 49](#_Toc207267322)

[Bảng 4.15 Mô tả sơ đồ tuần tự: Đặt hàng 50](#_Toc207267323)

[Bảng 4.16 Mô tả sơ đồ tuần tự: Đổi điểm thưởng lấy Voucher 51](#_Toc207267324)

[Bảng 4.17 Mô tả sơ tuần tự: Gửi đánh giá đơn hàng 53](#_Toc207267325)

[Bảng 4.18 Mô tả sơ đồ tuần tự: Thêm sản phẩm vào giỏ hàng 54](#_Toc207267326)

[Bảng 4.19 Mô tả sơ đồ tuần tự: Tìm kiếm và Xem chi tiết sản phẩm 56](#_Toc207267327)

[Bảng 4.20 Mô tả sơ đồ tuần tự: Admin cập nhật trạng thái đơn hàng 57](#_Toc207267328)

[Bảng 4.21 Mô tả sơ đồ tuần tự: Admin gán nhãn hiệu hàng loạt 58](#_Toc207267329)

[Bảng 4.22 Mô tả sơ đồ tuần tự: Admin nhập kho hàng loạt từ Excel 60](#_Toc207267330)

[Bảng 4.23 Mô tả sơ đồ tuần tự: Admin phản hồi bình luận 61](#_Toc207267331)

[Bảng 4.24 Mô tả sơ đồ tuần tự: Admin cập nhật vai trò người dùng 62](#_Toc207267332)

[Bảng 4.25 Mô tả sơ đồ tuần tự: Admin quản lý nội dung (Thêm/Sửa) 63](#_Toc207267333)

[Bảng 4.26 Mô tả sơ đồ tuần tự: Admin thêm sản phẩm mới 64](#_Toc207267334)

**DANH SÁCH HÌNH ẢNH**

[Hình 3.1 Sơ đồ Quan hệ Thực thể (ERD) 16](#_Toc207267226)

[Hình 3.2 Sơ đồ lớp (Class Diagram) 23](#_Toc207267227)

[Hình 4.1 Sơ đồ Usecase tổng quát 25](#_Toc207267228)

[Hình 4.2 Sơ đồ hoạt động: Đăng ký và đăng nhập 27](#_Toc207267229)

[Hình 4.3 Sơ đồ hoạt động: Đổi điểm thưởng lấy Voucher 29](#_Toc207267230)

[Hình 4.4 Sơ đồ hoạt động: Gửi đánh giá sản phẩm 31](#_Toc207267231)

[Hình 4.5 Sơ đồ hoạt động: Mua hàng của khách hàng 33](#_Toc207267232)

[Hình 4.6 Sơ đồ hoạt động: Quản lý danh sách yêu thích 35](#_Toc207267233)

[Hình 4.7 Sơ đồ hoạt động: quản lý tài khoản cá nhân 36](#_Toc207267234)

[Hình 4.8 Sơ đồ hoạt động: Nhập kho hàng loạt từ Excel 37](#_Toc207267235)

[Hình 4.9 Sơ đồ hoạt động: Xử lý đơn hàng 39](#_Toc207267236)

[Hình 4.10 Sơ đồ hoạt động: Quản lý & Phân quyền người dùng 40](#_Toc207267237)

[Hình 4.11 Sơ đồ hoạt động: Quản lý nội dung (CMS) 42](#_Toc207267238)

[Hình 4.12 Sơ đồ hoạt động: Quản lý sản phẩm & Gán nhãn hiệu hàng loạt 44](#_Toc207267239)

[Hình 4.13 Sơ đồ hoạt động: Thêm sản phẩm mới 46](#_Toc207267240)

[Hình 4.14 Sơ đồ tuần tự: Cập nhật thông cá nhân 48](#_Toc207267241)

[Hình 4.15 Sơ đồ tuần tự: Đăng nhập người dùng 49](#_Toc207267242)

[Hình 4.16 Sơ đồ tuần tự: Đặt hàng 50](#_Toc207267243)

[Hình 4.17 Sơ đồ tuần tự: Đổi điểm thưởng lấy Voucher 51](#_Toc207267244)

[Hình 4.18 Sơ tuần tự: Gửi đánh giá đơn hàng 53](#_Toc207267245)

[Hình 4.19 Sơ đồ tuần tự: Thêm sản phẩm vào giỏ hàng 54](#_Toc207267246)

[Hình 4.20 Sơ đồ tuần tự: Tìm kiếm và Xem chi tiết sản phẩm 56](#_Toc207267247)

[Hình 4.21 Sơ đồ tuần tự: Admin cập nhật trạng thái đơn hàng 57](#_Toc207267248)

[Hình 4.22 Sơ đồ tuần tự: Admin gán nhãn hiệu hàng loạt 58](#_Toc207267249)

[Hình 4.23 Sơ đồ tuần tự: Admin nhập kho hàng loạt từ Excel 59](#_Toc207267250)

[Hình 4.24 Sơ đồ tuần tự: Admin phản hồi bình luận 61](#_Toc207267251)

[Hình 4.25 Sơ đồ tuần tự: Admin cập nhật vai trò người dùng 62](#_Toc207267252)

[Hình 4.26 Sơ đồ tuần tự: Admin quản lý nội dung (Thêm/Sửa) 63](#_Toc207267253)

[Hình 4.27 Sơ đồ tuần tự: Admin thêm sản phẩm mới 64](#_Toc207267254)

[Hình 4.28 Trang chủ (HomePage) và Hiển thị Sản phẩm 65](#_Toc207267255)

[Hình 4.29 Trang giỏ hàng (CartPage) 66](#_Toc207267256)

[Hình 4.30 Trang thanh toán (CheckoutPage) 67](#_Toc207267257)

[Hình 4.31 Trang tài khoản Người dùng (ProfilePage) 68](#_Toc207267258)

[Hình 4.32 Trang lịch sử đơn hàng 69](#_Toc207267259)

[Hình 4.33 Trang danh sách yêu thích 69](#_Toc207267260)

[Hình 4.34 Trang danh sách địa chỉ 70](#_Toc207267261)

[Hình 4.35 Trang cài đặt 70](#_Toc207267262)

[Hình 4.36 Trang đổi điểm thưởng 71](#_Toc207267263)

[Hình 4.37 Trang quản lý sản phẩm 72](#_Toc207267264)

[Hình 4.38 Trang quản lý danh mục 72](#_Toc207267265)

[Hình 4.39 Trang quản lý chi nhánh 73](#_Toc207267266)

[Hình 4.40 Trang quản lý nhãn hiệu 73](#_Toc207267267)

[Hình 4.41 Trang quản lý nội dung 74](#_Toc207267268)

[Hình 4.42 Trang quản lý nhập kho 74](#_Toc207267269)

# MỞ ĐẦU

## 1. Lý do chọn đề tài

Trong bối cảnh cuộc cách mạng công nghiệp 4.0 đang diễn ra mạnh mẽ, thương mại điện tử (TMĐT) đã và đang trở thành một xu hướng tất yếu, len lỏi vào mọi ngóc ngách của đời sống kinh tế - xã hội. Đặc biệt, sau những biến động toàn cầu như đại dịch COVID-19, thói quen mua sắm của người tiêu dùng đã có sự dịch chuyển rõ rệt từ các kênh truyền thống sang nền tảng trực tuyến. Trong số các ngành hàng, thực phẩm và hàng tiêu dùng nhanh (FMCG) là một trong những lĩnh vực chứng kiến sự tăng trưởng bùng nổ nhất trên không gian mạng. Người tiêu dùng ngày càng ưu tiên sự tiện lợi, an toàn và khả năng truy xuất nguồn gốc sản phẩm, tạo ra một nhu cầu lớn cho các nền tảng TMĐT chuyên biệt về nông sản và thực phẩm sạch.

Tuy nhiên, thị trường này cũng đặt ra nhiều thách thức không nhỏ: làm thế nào để quản lý hiệu quả một lượng lớn sản phẩm với vòng đời ngắn? Làm sao để tối ưu hóa quy trình vận hành từ tồn kho, đặt hàng đến giao nhận? Và quan trọng hơn, làm thế nào để xây dựng một trải nghiệm người dùng liền mạch, tin cậy, đủ sức cạnh tranh với các "ông lớn" trên thị trường?

Nhận thấy tiềm năng cũng như những thách thức đó, đề tài **"Xây dựng Website Thương mại điện tử FreshFood - Nền tảng Nông sản và Hàng tiêu dùng"** được thực hiện với mục tiêu không chỉ tạo ra một trang web bán hàng đơn thuần, mà còn xây dựng một hệ thống quản trị toàn diện, linh hoạt, có khả năng mở rộng trong tương lai. Đồ án này là cơ hội để áp dụng các kiến thức đã học về TMĐT, lập trình web và quản trị hệ thống vào một dự án thực tế, giải quyết một bài toán kinh doanh cụ thể và có giá trị.

## 2. Mục tiêu của đề tài

Đồ án hướng tới việc hoàn thành các mục tiêu chính sau:

* **Xây dựng một giao diện người dùng (Client-side) hoàn chỉnh:** Tạo ra một trang web TMĐT thân thiện, dễ sử dụng cho khách hàng, với đầy đủ các chức năng cốt lõi: xem sản phẩm, tìm kiếm, lọc, giỏ hàng, thanh toán, quản lý tài khoản cá nhân, và các tính năng tương tác như đánh giá sản phẩm, danh sách yêu thích.
* **Xây dựng một hệ thống quản trị (Admin Panel) mạnh mẽ:** Phát triển một khu vực quản trị nội bộ cho phép quản lý toàn diện các khía cạnh của website, bao gồm: quản lý sản phẩm, danh mục, đơn hàng, khách hàng, chương trình khuyến mãi, và các nội dung động khác.
* **Triển khai các mô hình kinh doanh phức tạp:**
  + **Mô hình đa chi nhánh (Multi-branch):** Cho phép hệ thống quản lý tồn kho và vận hành theo nhiều chi nhánh địa lý khác nhau, tối ưu hóa quá trình giao vận.
  + **Nền tảng cho mô hình đa nhà cung cấp (Multi-vendor Marketplace):** Xây dựng cấu trúc dữ liệu và hệ thống phân quyền có khả năng mở rộng thành một sàn TMĐT, nơi nhiều cửa hàng có thể đăng ký và bán sản phẩm của họ.
* **Áp dụng công nghệ hiện đại:** Sử dụng ReactJS để xây dựng giao diện người dùng linh hoạt và Google Firebase làm nền tảng Backend-as-a-Service (BaaS) để quản lý cơ sở dữ liệu, xác thực người dùng và các quy tắc bảo mật một cách hiệu quả và tiết kiệm chi phí.

## 3. Đối tượng và phạm vi nghiên cứu

* **Đối tượng nghiên cứu:** Các quy trình, mô hình hoạt động và công nghệ liên quan đến việc xây dựng và vận hành một website TMĐT, đặc biệt trong lĩnh vực bán lẻ thực phẩm và hàng tiêu dùng.
* **Phạm vi nghiên cứu:**
  + **Về chức năng:** Đồ án tập trung vào việc xây dựng các module chính của một hệ thống TMĐT: quản lý người dùng, sản phẩm, danh mục, giỏ hàng, thanh toán, đơn hàng, khuyến mãi, đánh giá, chi nhánh và nhà cung cấp.
  + **Về công nghệ:** Sử dụng ReactJS cho Frontend và bộ công cụ của Google Firebase (Firestore, Authentication) cho Backend.
  + **Giới hạn:** Đồ án không đi sâu vào các vấn đề như tích hợp cổng thanh toán thực tế của bên thứ ba, các thuật toán gợi ý sản phẩm phức tạp (machine learning), hay các chiến dịch marketing quy mô lớn.

## 4. Bố cục của báo cáo

Ngoài phần Mở đầu và Kết luận, báo cáo được cấu trúc thành 5 chương chính:

* **Chương 1: Tổng quan tình hình nghiên cứu:** Phân tích bối cảnh thị trường TMĐT tại Việt Nam, xu hướng mua sắm thực phẩm trực tuyến và các công nghệ phổ biến được áp dụng.
* **Chương 2: Cơ sở lý luận:** Trình bày các khái niệm, mô hình lý thuyết về TMĐT, kiến trúc hệ thống và các công nghệ nền tảng được sử dụng trong đồ án (React, Firebase).
* **Chương 3: Cơ sở sở liệu:** Thiết kế và phân tích chi tiết cấu trúc cơ sở dữ liệu NoSQL (Firestore) được sử dụng để lưu trữ toàn bộ dữ liệu của hệ thống.
* **Chương 4: Phân tích và xây dựng hệ thống:** Đây là chương trọng tâm, mô tả chi tiết quá trình xây dựng các module chức năng của website FreshFood, từ giao diện khách hàng đến hệ thống quản trị cho Admin và nhà cung cấp.
* **Chương 5: Kết luận và hướng phát triển hệ thống:** Tổng kết những kết quả đã đạt được, đánh giá ưu nhược điểm của hệ thống và đề xuất các hướng phát triển, nâng cấp trong tương lai.

# CHƯƠNG 1: TỔNG QUAN TÌNH HÌNH NGHIÊN CỨU

## 1.1. Bối cảnh phát triển Thương mại điện tử tại Việt Nam

Thị trường thương mại điện tử Việt Nam trong những năm gần đây đã chứng kiến một sự phát triển vượt bậc, trở thành một trong những thị trường có tốc độ tăng trưởng nhanh nhất khu vực Đông Nam Á. Theo Sách trắng Thương mại điện tử Việt Nam, quy mô thị trường bán lẻ điện tử (B2C) đã liên tục tăng trưởng ở mức hai con số, thu hút sự tham gia của hàng triệu người tiêu dùng và hàng chục nghìn doanh nghiệp.

Sự phát triển này được thúc đẩy bởi nhiều yếu tố cộng hưởng:

* **Tỷ lệ thâm nhập Internet và Smartphone cao:** Với dân số trẻ, năng động và tỷ lệ người dùng Internet, smartphone ngày càng tăng, Việt Nam có một nền tảng vững chắc cho sự phát triển của các dịch vụ trực tuyến.
* **Sự thay đổi trong hành vi người tiêu dùng:** Người tiêu dùng, đặc biệt là thế hệ Millennials và Gen Z, ngày càng quen thuộc và ưa chuộng sự tiện lợi, nhanh chóng của mua sắm online. Họ có xu hướng tìm kiếm thông tin, so sánh giá cả và đọc đánh giá sản phẩm trên mạng trước khi ra quyết định.
* **Sự đầu tư mạnh mẽ từ các doanh nghiệp:** Các "ông lớn" trong và ngoài nước như Shopee, Lazada, Tiki, a... liên tục đổ vốn đầu tư vào hạ tầng, logistics, và các chiến dịch marketing rầm rộ, góp phần giáo dục thị trường và tạo dựng niềm tin cho người tiêu dùng.
* **Chính sách hỗ trợ từ Chính phủ:** Chính phủ Việt Nam đã ban hành nhiều chính sách và kế hoạch tổng thể nhằm thúc đẩy kinh tế số và TMĐT, tạo ra một hành lang pháp lý ngày càng hoàn thiện.

## 1.2. Xu hướng mua sắm thực phẩm và hàng tiêu dùng trực tuyến

Trong bức tranh chung của TMĐT, ngành hàng thực phẩm và hàng tiêu dùng nhanh (FMCG), hay còn gọi là e-grocery, nổi lên như một điểm sáng với tốc độ tăng trưởng ấn tượng nhất. Nếu trước đây, người tiêu dùng còn e dè trong việc mua thực phẩm tươi sống qua mạng do lo ngại về chất lượng và thời gian giao hàng, thì những rào cản này đang dần được xóa bỏ.

Các yếu tố chính thúc đẩy xu hướng này bao gồm:

* **Nhu cầu về sự tiện lợi:** Cuộc sống bận rộn khiến người tiêu dùng, đặc biệt ở các đô thị lớn, có ít thời gian hơn cho việc đi chợ hay siêu thị truyền thống. Việc đặt hàng online và được giao tận nhà giúp họ tiết kiệm thời gian và công sức.
* **Mối quan tâm về an toàn và sức khỏe:** Người tiêu dùng ngày càng quan tâm đến nguồn gốc xuất xứ và chất lượng của thực phẩm. Các nền tảng TMĐT chuyên về nông sản sạch, có chứng nhận VietGAP, GlobalGAP... đáp ứng tốt nhu cầu này bằng cách cung cấp thông tin minh bạch và cam kết chất lượng.
* **Sự phát triển của hạ tầng logistics:** Sự ra đời của các dịch vụ giao hàng nhanh (instant delivery) với đội ngũ shipper đông đảo và công nghệ tối ưu hóa lộ trình đã giải quyết được bài toán giao hàng thực phẩm tươi sống trong thời gian ngắn, đảm bảo sản phẩm vẫn giữ được độ tươi ngon.
* **Sự đa dạng của các mô hình kinh doanh:** Thị trường không chỉ có các siêu thị lớn (Co.opmart, Big C) tham gia, mà còn có sự xuất hiện của các mô hình chuyên biệt như "đi chợ hộ" (GrabMart, Shopee Food), các cửa hàng thực phẩm hữu cơ online, và các nền tảng kết nối trực tiếp từ nông trại đến người tiêu dùng.

Dự án FreshFood được xây dựng để đón đầu xu hướng này, tập trung vào việc cung cấp một nền tảng chuyên biệt cho nông sản và hàng tiêu dùng, giải quyết các nhu cầu cụ thể của ngành hàng này.

## 1.3. Các công nghệ phổ biến trong xây dựng Website TMĐT hiện đại

Để xây dựng một website TMĐT hiệu quả, việc lựa chọn công nghệ phù hợp là yếu tố then chốt. Hiện nay, các xu hướng công nghệ chính bao gồm:

* **Kiến trúc Headless Commerce:** Tách biệt phần giao diện người dùng (Frontend - "cái đầu") và phần xử lý logic, dữ liệu (Backend - "cái thân"). Điều này cho phép doanh nghiệp linh hoạt trong việc xây dựng nhiều giao diện khác nhau (web, app di động, thiết bị IoT) mà vẫn sử dụng chung một hệ thống backend.
* **Single Page Application (SPA):** Các ứng dụng trang đơn như dự án FreshFood mang lại trải nghiệm người dùng mượt mà, nhanh chóng như sử dụng một ứng dụng trên máy tính. Thay vì phải tải lại toàn bộ trang mỗi khi người dùng thực hiện một hành động, SPA chỉ tải các phần dữ liệu cần thiết, giúp tăng tốc độ và giảm tải cho máy chủ. Các thư viện/framework phổ biến để xây dựng SPA là **ReactJS**, Angular và Vue.js.
* **Backend-as-a-Service (BaaS):** Các nền tảng như **Google Firebase**, AWS Amplify cung cấp các dịch vụ backend đã được xây dựng sẵn (cơ sở dữ liệu, xác thực, lưu trữ file...). Điều này giúp các nhà phát triển, đặc biệt là các đội nhóm nhỏ hoặc cá nhân, có thể xây dựng các ứng dụng phức tạp một cách nhanh chóng mà không cần tốn nhiều công sức quản lý hạ tầng máy chủ.
* **Progressive Web App (PWA):** Kết hợp những ưu điểm tốt nhất của website và ứng dụng di động. PWA cho phép người dùng "cài đặt" trang web lên màn hình chính của điện thoại, gửi thông báo đẩy (push notifications) và có thể hoạt động offline.

Dự án FreshFood đã lựa chọn **ReactJS** và **Firebase**, hai công nghệ tiêu biểu cho xu hướng SPA và BaaS, để tận dụng tối đa các ưu điểm về tốc độ phát triển, khả năng mở rộng và trải nghiệm người dùng hiện đại.

# CHƯƠNG 2: CƠ SỞ LÝ LUẬN

## 2.1. Tổng quan về Thương mại điện tử (E-commerce)

### 2.1.1. Khái niệm

Thương mại điện tử (E-commerce) là hình thức kinh doanh, mua bán hàng hóa và dịch vụ thông qua mạng Internet và các phương tiện điện tử. Nó bao gồm tất cả các hoạt động từ quảng cáo, tiếp thị, đặt hàng, thanh toán đến hỗ trợ sau bán hàng, tất cả đều được thực hiện trên một nền tảng số.

### 2.1.2. Các mô hình TMĐT phổ biến

TMĐT có thể được phân loại thành nhiều mô hình khác nhau, dựa trên đối tượng tham gia giao dịch:

* **B2C (Business-to-Consumer):** Doanh nghiệp bán hàng trực tiếp cho người tiêu dùng cuối cùng. Đây là mô hình phổ biến nhất, mà dự án FreshFood là một ví dụ điển hình.
* **B2B (Business-to-Business):** Doanh nghiệp bán hàng cho các doanh nghiệp khác.
* **C2C (Consumer-to-Consumer):** Người tiêu dùng bán hàng cho nhau, thường thông qua một nền tảng trung gian (ví dụ: Chợ Tốt, các nhóm mua bán trên Facebook).
* **C2B (Consumer-to-Business):** Người tiêu dùng cung cấp sản phẩm hoặc dịch vụ cho doanh nghiệp.
* **Marketplace (Sàn giao dịch TMĐT):** Một nền tảng trung gian kết nối nhiều người bán và người mua (ví dụ: Shopee, Tiki). Dự án FreshFood được xây dựng với cấu trúc nền tảng có thể mở rộng thành mô hình này trong tương lai.

### 2.1.3. Các thành phần chính của một Website TMĐT

Một hệ thống TMĐT B2C hoàn chỉnh thường bao gồm các module chức năng cốt lõi sau:

* **Khu vực Khách hàng (Client-facing):**
  + **Trang chủ:** Cửa ngõ của website, trưng bày các sản phẩm nổi bật, khuyến mãi.
  + **Danh mục sản phẩm (Product Catalog):** Hiển thị danh sách sản phẩm, cho phép lọc, sắp xếp và tìm kiếm.
  + **Trang chi tiết sản phẩm:** Cung cấp thông tin đầy đủ, hình ảnh, mô tả và đánh giá.
  + **Giỏ hàng (Shopping Cart):** Cho phép người dùng thêm, xóa, cập nhật số lượng sản phẩm.
  + **Thanh toán (Checkout):** Quy trình nhập thông tin giao hàng, chọn phương thức thanh toán và hoàn tất đơn hàng.
  + **Tài khoản người dùng:** Quản lý thông tin cá nhân, địa chỉ, xem lịch sử đơn hàng.
* **Khu vực Quản trị (Admin Panel):**
  + **Quản lý Sản phẩm:** CRUD (Tạo, Đọc, Sửa, Xóa) sản phẩm.
  + **Quản lý danh mục:** CRUD (Tạo, Đọc, Sửa, Xóa) danh mục
  + **Quản lý Đơn hàng:** Xem và cập nhật trạng thái đơn hàng.
  + **Quản lý Khách hàng:** Xem thông tin và lịch sử mua hàng.
  + **Quản lý Nội dung:** Chỉnh sửa các trang thông tin như FAQ, chính sách.
  + **Báo cáo & Thống kê:** Theo dõi các chỉ số kinh doanh.

## 2.2. Công nghệ sử dụng trong đồ án

### 2.2.1. ReactJS - Thư viện xây dựng giao diện người dùng

ReactJS là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook, chuyên dùng để xây dựng giao diện người dùng (UI). Các đặc điểm chính giúp ReactJS trở nên phổ biến bao gồm:

* **Kiến trúc dựa trên Component:** Giao diện được chia thành các thành phần (component) nhỏ, độc lập và có thể tái sử dụng (ví dụ: Header, Footer, ProductCard). Điều này giúp mã nguồn trở nên gọn gàng, dễ quản lý và bảo trì.
* **Virtual DOM (DOM ảo):** React sử dụng một bản sao của DOM (Document Object Model) trong bộ nhớ. Khi có sự thay đổi về trạng thái, React sẽ so sánh Virtual DOM mới với bản cũ, tính toán sự khác biệt và chỉ cập nhật những phần thực sự thay đổi trên DOM thật. Điều này giúp tối ưu hóa hiệu năng và làm cho các ứng dụng SPA chạy rất nhanh.
* **Luồng dữ liệu một chiều (One-way Data Flow):** Dữ liệu trong React thường được truyền từ component cha xuống component con, giúp cho việc kiểm soát và gỡ lỗi trở nên dễ dàng hơn.
* **Hệ sinh thái rộng lớn:** React có một cộng đồng phát triển đông đảo, cung cấp hàng ngàn thư viện và công cụ hỗ trợ (ví dụ: React Router cho điều hướng, Redux cho quản lý trạng thái).

Trong dự án FreshFood, ReactJS được sử dụng để xây dựng toàn bộ giao diện người dùng, từ trang khách hàng đến trang quản trị, tạo ra một trải nghiệm ứng dụng trang đơn (SPA) mượt mà và linh hoạt.

### 2.2.2. Google Firebase - Nền tảng Backend-as-a-Service (BaaS)

Firebase là một nền tảng phát triển ứng dụng di động và web của Google, cung cấp một bộ công cụ mạnh mẽ giúp xây dựng backend một cách nhanh chóng. Các dịch vụ chính của Firebase được sử dụng trong dự án FreshFood bao gồm:

* **Firestore:** Là một cơ sở dữ liệu NoSQL, linh hoạt, và thời gian thực (real-time).
  + **NoSQL:** Dữ liệu được lưu trữ dưới dạng các document (tương tự JSON) được gom nhóm trong các collection, thay vì các bảng có cấu trúc cố định như cơ sở dữ liệu quan hệ (SQL). Điều này giúp cho việc phát triển và thay đổi cấu trúc dữ liệu trở nên rất linh hoạt.
  + **Real-time:** Khi có bất kỳ thay đổi nào trong cơ sở dữ liệu, Firestore sẽ tự động đẩy dữ liệu mới xuống tất cả các client đang lắng nghe, giúp cho các tính năng như giỏ hàng, trạng thái đơn hàng được cập nhật ngay lập tức mà không cần người dùng phải tải lại trang.
* **Firebase Authentication:** Cung cấp dịch vụ xác thực người dùng hoàn chỉnh, hỗ trợ nhiều phương thức đăng nhập như Email/Mật khẩu, Google, Facebook... Nó giúp đơn giản hóa việc quản lý tài khoản, mật khẩu và bảo mật cho người dùng.
* **Firebase Hosting:** Dịch vụ lưu trữ và triển khai website tĩnh và ứng dụng web một cách nhanh chóng và an toàn, với CDN toàn cầu.
* **(Tùy chọn) Cloud Functions:** Cho phép viết và chạy các đoạn code backend trên máy chủ của Google mà không cần quản lý server. Đây là công cụ cần thiết cho các tác vụ cần bảo mật cao hoặc xử lý nền, như xử lý thanh toán, gửi email thông báo, hoặc tự động cộng điểm thưởng.

Việc lựa chọn Firebase giúp dự án FreshFood có thể tập trung vào việc xây dựng logic nghiệp vụ và trải nghiệm người dùng mà không cần tốn quá nhiều thời gian và chi phí cho việc thiết lập và bảo trì hạ tầng backend.

### 2.2.3 Vercel Serverless Functions - Môi trường Backend linh hoạt

Vercel Serverless Functions là một giải pháp cho phép chạy mã nguồn backend theo yêu cầu mà không cần phải thiết lập hay quản lý một máy chủ truyền thống. Trong đồ án FreshFood, công nghệ này đóng một vai trò cực kỳ quan trọng trong việc xử lý các tác vụ nhạy cảm và cần bảo mật cao mà không thể thực hiện ở phía client (trình duyệt). Cụ thể, nó được sử dụng để làm trung gian giao tiếp an toàn với API của cổng thanh toán Stripe. Khi người dùng thực hiện thanh toán, giao diện React sẽ gửi yêu cầu đến một Vercel Function; function này sau đó sẽ gọi đến Stripe bằng "Secret Key" được lưu trữ an toàn trên máy chủ của Vercel, tạo ra một phiên giao dịch và gửi lại mã giao dịch tạm thời cho client để hoàn tất thanh toán. Cách làm này đảm bảo các thông tin nhạy cảm không bao giờ bị lộ ra bên ngoài.

# CHƯƠNG 3: CƠ SỞ DỮ LIỆU

Hệ thống FreshFood sử dụng **Cloud Firestore**, một cơ sở dữ liệu NoSQL của Google Firebase, để lưu trữ toàn bộ dữ liệu. Cấu trúc dữ liệu được thiết kế theo hướng linh hoạt, dễ mở rộng, bao gồm các collection (bộ sưu tập) chính sau đây.

## 3.1. Sơ đồ Quan hệ Thực thể (ERD)

Mặc dù Firestore là NoSQL, việc hình dung mối quan hệ giữa các collection là rất quan trọng để hiểu được luồng dữ liệu của hệ thống.

* **Một-Nhiều (One-to-Many):**
  + Một User có thể có nhiều Order, nhiều Address.
  + Một Category có thể có nhiều Product.
  + Một Product có thể có nhiều Review, nhiều Inventory (tồn kho).
  + Một Branch có thể có nhiều Inventory.
* **Nhiều-Nhiều (Many-to-Many - thông qua subcollection hoặc trường array):**
  + Một User có thể yêu thích nhiều Product (thông qua subcollection wishlist).
  + Một User có thể có nhiều Promotion (mã giảm giá).

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 3.0.1 Sơ đồ Quan hệ Thực thể (ERD)

Bảng 3.0.1 Mô tả sơ đồ quan hệ thực thể (ERD)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thực thể (Entity) | Mô tả | Các thuộc tính (Attributes) | Mối quan hệ (Relationships) |
| 1 | Users (Người dùng) | Lưu trữ thông tin tài khoản của người dùng (khách hàng, nhân viên, quản trị viên). | • userId (PK): Mã người dùng  • displayName: Tên hiển thị  • email: Email  • role: Vai trò | • (1-N) với Orders: Một người dùng có thể đặt nhiều đơn hàng.  • (1-N) với Promotions: Một người dùng (admin) tạo nhiều khuyến mãi.  • (1-N) với Purchases: Một người dùng (nhân viên) tạo nhiều phiếu nhập hàng. |
| 2 | Products (Sản phẩm) | Chứa thông tin chi tiết về các sản phẩm được kinh doanh. | • productId (PK): Mã sản phẩm  • name: Tên sản phẩm  • description: Mô tả  • price: Giá bán  • categoryId (FK): Mã phân loại  • brandId (FK): Mã thương hiệu | • (N-1) với Categories: Nhiều sản phẩm thuộc một phân loại.  • (N-1) với Brands: Nhiều sản phẩm thuộc một thương hiệu.  • (1-N) với Inventory: Một sản phẩm có thể có tồn kho ở nhiều chi nhánh. |
| 3 | Categories (Phân loại) | Dùng để phân nhóm các sản phẩm. Hỗ trợ cấu trúc danh mục đa cấp. | • categoryId (PK): Mã phân loại  • name: Tên phân loại  • parentId (FK): Mã danh mục cha  • description: Mô tả | • (1-N) với Products: Một phân loại chứa nhiều sản phẩm. |
| 4 | Brands (Thương hiệu) | Quản lý thông tin các thương hiệu của sản phẩm. | • brandId (PK): Mã thương hiệu  • brandName: Tên thương hiệu | • (1-N) với Products: Một thương hiệu có nhiều sản phẩm. |
| 5 | Branches (Chi nhánh) | Quản lý thông tin các chi nhánh, cửa hàng kinh doanh. | • branchId (PK): Mã chi nhánh  • branchName: Tên chi nhánh  • city: Thành phố  • deliveryDistricts: Khu vực giao hàng | • (1-N) với Orders: Một chi nhánh xử lý nhiều đơn hàng.  • (1-N) với Inventory: Một chi nhánh có tồn kho của nhiều sản phẩm. |
| 6 | Inventory (Tồn kho) | Thực thể liên kết, quản lý số lượng tồn kho của một sản phẩm cụ thể tại một chi nhánh cụ thể. | • productId (FK)  • branchId (FK)  • stock: Số lượng tồn kho Ghi chú: (productId, branchId) là khóa chính phức hợp. | • Giải quyết mối quan hệ N-N giữa Products và Branches. |
| 7 | Orders (Đơn hàng) | Lưu trữ thông tin các giao dịch mua hàng của khách. | • orderId (PK): Mã đơn hàng  • userId (FK)  • branchId (FK)  • promoId (FK)  • items: Chi tiết sản phẩm  • totalAmount: Tổng tiền  • status: Trạng thái  Đề xuất: Tách items thành bảng OrderDetails. | • (N-1) với Users: Nhiều đơn hàng thuộc về một người dùng.  • (N-1) với Branches: Nhiều đơn hàng được xử lý bởi một chi nhánh.  • (N-1) với Promotions: Nhiều đơn hàng có thể dùng chung một khuyến mãi. |
| 8 | Promotions (Khuyến mãi) | Quản lý các chương trình, mã giảm giá. | • promoId (PK): Mã khuyến mãi  • code: Mã code  • description: Mô tả  • discountType: Loại giảm giá  • userId (FK): Người tạo | • (N-1) với Users: Nhiều khuyến mãi được tạo bởi một admin.  • (1-N) với Orders: Một khuyến mãi có thể được áp dụng cho nhiều đơn hàng. |
| 9 | Purchases (Nhập hàng) | Ghi lại lịch sử các hoạt động nhập hàng từ nhà cung cấp. | • purchaseId (PK): Mã phiếu nhập  • items: Chi tiết hàng nhập.  • totalCost: Tổng chi phí  • supplier: Tên nhà cung cấp  Đề xuất: Tách items thành bảng PurchaseDetails và supplier thành thực thể Suppliers. | • (N-1) với Users: Nhiều phiếu nhập hàng được tạo bởi một nhân viên. |

## 3.2. Mô tả chi tiết các Collection

Hệ thống FreshFood sử dụng Cloud Firestore làm cơ sở dữ liệu chính. Dữ liệu được tổ chức thành các collection (bộ sưu tập), mỗi collection chứa các document (tài liệu) có cấu trúc linh hoạt. Dưới đây là mô tả chi tiết về cấu trúc của từng collection chính trong hệ thống.

**1. users**

* Mục đích: Lưu trữ thông tin tài khoản của tất cả người dùng, bao gồm khách hàng, nhân viên và quản trị viên.
* Document ID: userId (trùng với UID từ Firebase Authentication).
* Cấu trúc Document mẫu:
  + displayName (string): Tên hiển thị của người dùng (ví dụ: "Khang").
  + email (string): Địa chỉ email đăng nhập (ví dụ: "nhpk1412@gmail.com").
  + role (string): Vai trò của người dùng trong hệ thống, quyết định quyền hạn truy cập (customer, staff, admin).
  + createdAt (timestamp): Mốc thời gian khi tài khoản được tạo.
* Subcollections (Bộ sưu tập con):
  + addresses: Lưu các địa chỉ giao hàng của người dùng.
  + wishlist: Lưu danh sách các sản phẩm yêu thích.
  + cart: Lưu giỏ hàng, cho phép đồng bộ giữa các thiết bị.
  + pointsHistory: Lưu lịch sử chi tiết của các giao dịch điểm thưởng.

**2. products**

* Mục đích: Lưu trữ thông tin chi tiết về tất cả sản phẩm được bán trên trang web.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + name (string): Tên đầy đủ của sản phẩm (ví dụ: "Hộp 8 bánh trung thu Savoure Sài Gòn Bến Thành Vip 800g").
  + description (string): Mô tả chi tiết về sản phẩm.
  + price (number): Giá bán gốc của sản phẩm.
  + salePrice (number, tùy chọn): Giá bán sau khi đã áp dụng khuyến mãi.
  + onSale (boolean): Trạng thái cho biết sản phẩm có đang được khuyến mãi hay không.
  + imageUrls (array): Một mảng chứa các đường link (URL) đến hình ảnh của sản phẩm.
  + thumbnailUrl (string): URL của hình ảnh đại diện chính, hiển thị ở trang danh sách.
  + categoryId (string): ID tham chiếu đến document trong collection categories.
  + brandId (string): ID tham chiếu đến document trong collection brands.
  + purchaseCount (number): Tổng số lượt sản phẩm này đã được bán, dùng để sắp xếp "Bán chạy".
  + averageRating (number): Điểm đánh giá trung bình của sản phẩm.
  + reviewCount (number): Tổng số lượt đánh giá mà sản phẩm đã nhận.
  + createdAt (timestamp): Mốc thời gian sản phẩm được thêm vào hệ thống.
* Subcollections:
  + inventory: Quản lý tồn kho theo từng chi nhánh. Document ID là branchId, chứa trường { stock: number }.
  + reviews: Lưu trữ tất cả các đánh giá của khách hàng cho sản phẩm này.

**3. categories**

* Mục đích: Quản lý danh mục sản phẩm theo cấu trúc cha-con linh hoạt.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + name (string): Tên của danh mục (ví dụ: "Nước lẩu, viên thả lẩu").
  + parentId (string, tùy chọn): ID của danh mục cha. Nếu là null hoặc không có, đây là danh mục gốc.
  + createdAt (timestamp): Ngày tạo danh mục.

**4. brands**

* Mục đích: Quản lý thông tin các nhãn hiệu, thương hiệu có sản phẩm được bán trên website.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + brandName (string): Tên của nhãn hiệu (ví dụ: "Kinh Đô").
  + description (string): Thông tin mô tả chi tiết về nhãn hiệu.
  + logoUrl (string): Đường link đến logo của nhãn hiệu.

**5. orders**

* Mục đích: Lưu trữ thông tin chi tiết của tất cả các đơn hàng mà khách hàng đã đặt.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + userId (string): ID của người dùng đặt hàng.
  + branchId (string): ID của chi nhánh chịu trách nhiệm xử lý đơn hàng.
  + shippingInfo (map): Một object chứa thông tin giao hàng (name, phone, address).
  + items (array): Một mảng các object, mỗi object là một sản phẩm trong đơn hàng, chứa đầy đủ thông tin sản phẩm tại thời điểm mua (id, name, price, quantity...).
  + paymentMethod (string): Phương thức thanh toán được chọn (ví dụ: "STRIPE\_CARD", "COD").
  + paymentIntentId: (string, tùy chọn).
  + status (string): Trạng thái hiện tại của đơn hàng (ví dụ: "Đang xử lý", "Hoàn thành").
  + subtotal (number): Tổng tiền hàng trước khi giảm giá.
  + discount (number): Tổng số tiền được giảm giá.
  + totalAmount (number): Số tiền cuối cùng khách hàng phải trả.
  + createdAt (timestamp): Thời điểm đơn hàng được tạo.

**6. branches**

* Mục đích: Quản lý thông tin và các khu vực phục vụ của từng chi nhánh vật lý.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + branchName (string): Tên chi nhánh (ví dụ: "Chi nhánh TP.Hồ Chí Minh").
  + address (string): Địa chỉ cụ thể của chi nhánh.
  + city (string): Tên thành phố của chi nhánh.
  + deliveryDistricts (array): Một mảng chứa tên các quận/huyện mà chi nhánh này có thể giao hàng đến.

**7. purchases**

* Mục đích: Ghi lại lịch sử các lần nhập kho hàng hóa.
* Document ID: ID được Firestore tự động tạo.
* Cấu trúc Document mẫu:
  + supplier (string): Tên nhà cung cấp (ví dụ: "Nhập từ Excel").
  + items (array): Một mảng các object, mỗi object ghi lại thông tin một mặt hàng được nhập (productId, productName, quantity, purchasePrice).
  + totalCost (number): Tổng chi phí cho lần nhập kho.
  + createdAt (timestamp): Ngày nhập kho.

**8. promotions**

* Mục đích: Quản lý các mã giảm giá và chương trình khuyến mãi.
* Document ID: ID tự sinh hoặc chính là mã code.
* Cấu trúc Document mẫu:
  + code (string): Mã giảm giá mà người dùng sẽ nhập (ví dụ: "EVENT\_nKqy...").
  + description (string): Mô tả về chương trình khuyến mãi.
  + discountType (string): Loại giảm giá ("fixed" cho số tiền cố định, "percentage" cho phần trăm).
  + discountValue (number): Giá trị giảm.
  + expiresAt (timestamp): Thời điểm mã hết hạn.
  + userId (string, tùy chọn): Giới hạn mã chỉ cho một người dùng cụ thể.

**9. content**

* Mục đích: Quản lý nội dung động (CMS) cho các trang thông tin.
* Document ID: slug (URL thân thiện, ví dụ: "cach-thuc-dat-hang").
* Cấu trúc Document mẫu:
  + title (string): Tiêu đề của trang hoặc câu hỏi FAQ.
  + type (string): Loại nội dung ("faq", "policy", "blog").
  + content (string): Nội dung chi tiết của trang.
  + createdAt (timestamp): Ngày tạo.

## 3.3 Sơ đồ lớp (Class Diagram)

A diagram of a computer program

AI-generated content may be incorrect.

Hình 3.0.2 Sơ đồ lớp (Class Diagram)

# CHƯƠNG 4: PHÂN TÍCH VÀ XÂY DỰNG HỆ THỐNG

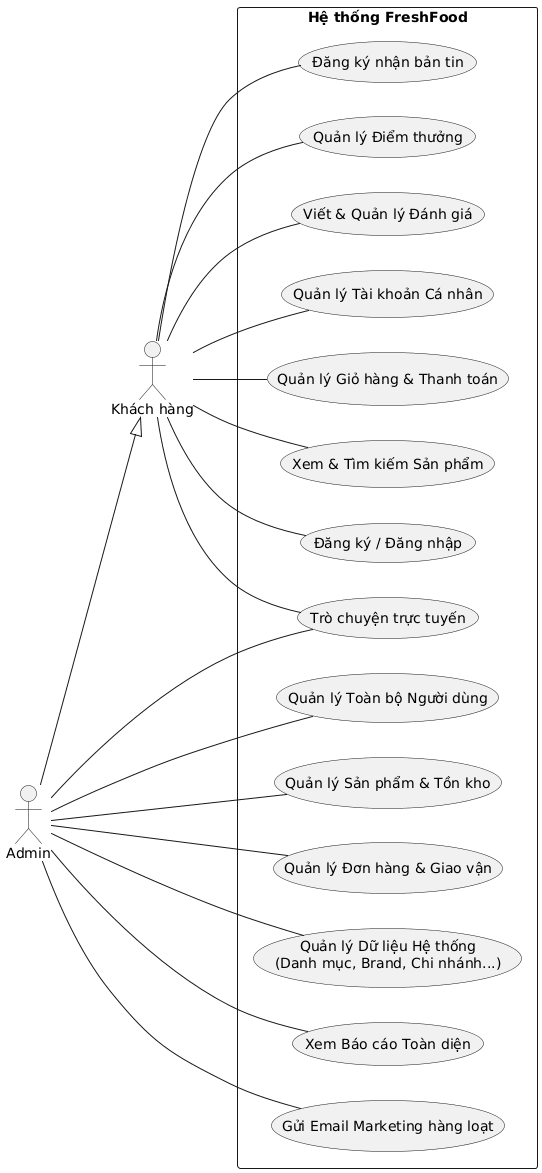
Chương này trình bày chi tiết về quá trình xây dựng các module chức năng của website FreshFood, dựa trên file mã nguồn App.js. Hệ thống được chia thành hai khu vực chính: Giao diện Khách hàng và Hệ thống Quản trị (Admin & Vendor).

## 4.1. Cấu trúc Tổng thể của Ứng dụng

Ứng dụng được xây dựng dưới dạng một **Single Page Application (SPA)**, với toàn bộ logic và các thành phần giao diện được quản lý trong một file duy nhất để tiện cho việc báo cáo.

* **Quản lý Trạng thái Toàn cục (Global State):** AppContext và AppProvider được sử dụng để cung cấp các dữ liệu và hàm dùng chung cho toàn bộ ứng dụng, như thông tin người dùng (user, userData), giỏ hàng (cart), chi nhánh đã chọn (selectedBranch), và hàm điều hướng (setPage). Đây là "trái tim" của ứng dụng, giúp các component có thể giao tiếp và chia sẻ dữ liệu một cách nhất quán.
* **Điều hướng (Routing):** Hệ thống sử dụng một cơ chế điều hướng tự xây dựng dựa trên state. Hàm renderPage() trong component App đóng vai trò như một bộ định tuyến, đọc giá trị của page từ AppContext để quyết định component nào sẽ được hiển thị cho người dùng. Cách làm này, dù đơn giản, nhưng đã mô phỏng được logic của các thư viện định tuyến chuyên nghiệp.

## 4.2 Sơ đồ Usecase tổng quát



Hình 4.0.1 Sơ đồ Usecase tổng quát

Sơ đồ xác định hai tác nhân (Actor) chính:

1. **Khách hàng**: Người dùng cuối thực hiện các hoạt động mua sắm. Các chức năng chính bao gồm:
   * Đăng ký, đăng nhập.
   * Tìm kiếm và xem thông tin sản phẩm.
   * Quản lý giỏ hàng và thực hiện thanh toán.
   * Quản lý thông tin tài khoản cá nhân.
   * Viết và quản lý các đánh giá sản phẩm.
   * Theo dõi điểm thưởng tích lũy.
   * Trò chuyện trực tuyến
2. **Admin**: Người quản trị, có toàn quyền vận hành hệ thống. Admin có thể thực hiện tất cả chức năng của **Khách hàng**, và có thêm các quyền quản lý chuyên sâu:
   * Quản lý tài khoản của tất cả người dùng.
   * Quản lý sản phẩm, danh mục, thương hiệu và tồn kho.
   * Xử lý đơn hàng và theo dõi giao vận.
   * Quản lý các dữ liệu lõi của hệ thống (chi nhánh, thương hiệu...).
   * Xem các báo cáo thống kê toàn diện.
   * Trò chuyện trực tuyến cùng khách hàng
   * Gửi email marketing cho khách hàng

## 4.3 Sơ đồ hoạt động

### 4.3.1 Sơ đồ hoạt động của khách hàng

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.2 Sơ đồ hoạt động: Đăng ký và đăng nhập

Bảng 4.0.1 Mô tả sơ đồ hoạt động: Đăng ký và đăng nhập

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Đăng ký & Đăng nhập | |
| **Tác nhân** | Người dùng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Đăng nhập thành công | 1. **Người dùng** vào trang web.  2. **Người dùng** nhấn nút "Đăng nhập".  3. **Hệ thống** hiển thị Modal.  4. **Người dùng** nhập Email và Mật khẩu chính xác.  5. **Người dùng** nhấn nút "Đăng nhập".  6. **Hệ thống** xác thực thông tin thành công và tải dữ liệu người dùng.  7. **Hệ thống** đóng Modal và cập nhật trạng thái đã đăng nhập. |
| Luồng phụ | Đăng ký tài khoản thành công | 1. **Người dùng** vào trang web và yêu cầu mở Modal Đăng nhập/Đăng ký.  2. **Người dùng** chọn tab "Đăng ký".  3. **Người dùng** điền đầy đủ Tên, Email, Mật khẩu hợp lệ.  4. **Người dùng** nhấn nút "Đăng ký".  5. **Hệ thống** tạo tài khoản thành công, tự động đăng nhập và lưu dữ liệu.  6. **Hệ thống** đóng Modal và cập nhật trạng thái đã đăng nhập. |
| Luồng ngoại lệ | Đăng nhập thất bại | 1. **Người dùng** nhập sai Email hoặc Mật khẩu.  2. **Người dùng** nhấn "Đăng nhập".  3. **Hệ thống** xác thực thất bại.  4. **Hệ thống** hiển thị thông báo lỗi và yêu cầu nhập lại. |
| Đăng ký thất bại | 1. **Người dùng** điền thông tin không hợp lệ (ví dụ: email đã tồn tại).  2. **Người dùng** nhấn "Đăng ký".  3. **Hệ thống** kiểm tra và xác định đăng ký không thành công.  4. **Hệ thống** hiển thị thông báo lỗi tương ứng. |

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.3 Sơ đồ hoạt động: Đổi điểm thưởng lấy Voucher

Bảng 4.0.2 Mô tả sơ đồ hoạt động: Đổi điểm thưởng lấy Voucher

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Đổi Điểm thưởng lấy Voucher | |
| **Tác nhân** | Khách hàng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Đổi điểm thành công | 1. **Khách hàng** truy cập trang "Hồ sơ của tôi" và chọn tab "Điểm thưởng".  2. **Hệ thống** hiển thị điểm hiện tại và các mốc quy đổi.  3. **Khách hàng** chọn một mốc đổi thưởng và nhấn nút "Đổi".  4. **Hệ thống** kiểm tra và xác nhận khách hàng có đủ điểm.  5. **Hệ thống** thực hiện giao dịch: trừ điểm của khách hàng, ghi lại lịch sử giao dịch, và tạo một mã voucher mới.  6. **Hệ thống** hiển thị thông báo "Đổi thành công!" kèm mã voucher cho khách hàng. |
| Luồng ngoại lệ | Không đủ điểm để đổi | 1. **Khách hàng** truy cập trang "Hồ sơ của tôi" và chọn tab "Điểm thưởng".  2. **Khách hàng** chọn một mốc đổi thưởng và nhấn nút "Đổi".  3. **Hệ thống** kiểm tra và xác định số điểm của khách hàng không đủ để thực hiện giao dịch.  4. **Hệ thống** hiển thị thông báo lỗi "Bạn không đủ điểm". |

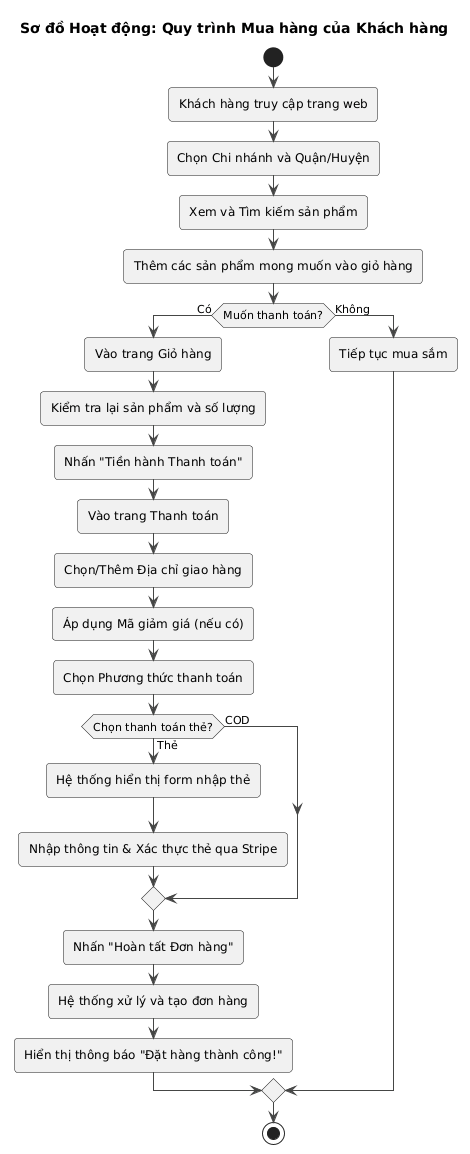
A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.4 Sơ đồ hoạt động: Gửi đánh giá sản phẩm

Bảng 4.0.3 Mô tả sơ đồ hoạt động: Gửi đánh giá sản phẩm

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Gửi Đánh giá Sản phẩm | |
| **Tác nhân** | Khách hàng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Gửi đánh giá thành công | 1. **Khách hàng** truy cập "Lịch sử Đơn hàng" và tìm một đơn hàng có trạng thái "Hoàn thành".  2. **Khách hàng** nhấn nút "Đánh giá" tại đơn hàng đó.  3. **Hệ thống** chuyển hướng đến trang đánh giá, hiển thị danh sách các sản phẩm cần đánh giá.  4. **Khách hàng** chọn sao (rating) và viết bình luận cho từng sản phẩm.  5. **Khách hàng** nhấn "Hoàn tất đánh giá" để gửi đi.  6. **Hệ thống** lưu lại nội dung đánh giá và cập nhật điểm xếp hạng trung bình của sản phẩm.  7. **Hệ thống** hiển thị thông báo cảm ơn và cập nhật trạng thái của đơn hàng thành "Đã đánh giá". |



Hình 4.0.5 Sơ đồ hoạt động: Mua hàng của khách hàng

Bảng 4.0.4 Mô tả sơ đồ hoạt động: Mua hàng của khách hàng

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quy trình Mua hàng của Khách hàng | |
| **Tác nhân** | Khách hàng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Hoàn tất đặt hàng (Thanh toán khi nhận hàng - COD) | 1. **Khách hàng** truy cập trang web, chọn chi nhánh và tìm kiếm sản phẩm.  2. **Khách hàng** thêm sản phẩm vào giỏ hàng và chọn tiến hành thanh toán.  3. Tại trang thanh toán, **Khách hàng** điền địa chỉ, áp dụng mã giảm giá (nếu có) và chọn phương thức thanh toán là "COD".  4. **Khách hàng** nhấn "Hoàn tất Đơn hàng".  5. **Hệ thống** xử lý, tạo đơn hàng mới và hiển thị thông báo "Đặt hàng thành công!". |
| Luồng phụ | Thanh toán bằng thẻ (qua Stripe) | 1. Tại bước chọn phương thức thanh toán, **Khách hàng** chọn "Thẻ".  2. **Hệ thống** hiển thị một form để nhập thông tin thẻ.  3. **Khách hàng** nhập thông tin và **Hệ thống** thực hiện xác thực thẻ thông qua dịch vụ bên thứ ba (Stripe).  4. Sau khi xác thực thẻ thành công, luồng sẽ tiếp tục với bước "Hoàn tất Đơn hàng". |
| Tiếp tục mua sắm | 1. Sau khi thêm một sản phẩm vào giỏ hàng, **Khách hàng** lựa chọn "Không" thanh toán ngay.  2. **Hệ thống** giữ nguyên các sản phẩm trong giỏ và luồng hoạt động quay trở lại bước "Xem và Tìm kiếm sản phẩm" để khách hàng có thể tiếp tục mua sắm. |

A diagram of a project

AI-generated content may be incorrect.

Hình 4.0.6 Sơ đồ hoạt động: Quản lý danh sách yêu thích

Bảng 4.0.5 Mô tả sơ đồ hoạt động: Quản lý danh sách yêu thích

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quản lý Danh sách Yêu thích | |
| **Tác nhân** | Khách hàng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Thêm sản phẩm vào danh sách yêu thích | 1. **Khách hàng** (đã đăng nhập) đang xem một sản phẩm chưa có trong danh sách.  2. **Khách hàng** nhấn vào biểu tượng trái tim (Heart icon).  3. **Hệ thống** kiểm tra và xác nhận sản phẩm chưa tồn tại trong danh sách.  4. **Hệ thống** thêm sản phẩm vào danh sách yêu thích của khách hàng.  5. **Hệ thống** cập nhật giao diện, biểu tượng trái tim được tô đầy. |
| Luồng phụ | Xóa sản phẩm khỏi danh sách yêu thích | 1. **Khách hàng** (đã đăng nhập) đang xem một sản phẩm đã có trong danh sách.  2. **Khách hàng** nhấn vào biểu tượng trái tim (đang được tô đầy).  3. **Hệ thống** kiểm tra và xác nhận sản phẩm đã tồn tại trong danh sách.  4. **Hệ thống** xóa sản phẩm khỏi danh sách yêu thích của khách hàng.  5. **Hệ thống** cập nhật giao diện, biểu tượng trái tim trở nên trống. |
| Luồng ngoại lệ | Người dùng chưa đăng nhập | 1. **Khách hàng** (chưa đăng nhập) xem một sản phẩm và nhấn vào biểu tượng trái tim.  2. **Hệ thống** kiểm tra và xác định người dùng chưa đăng nhập.  3. **Hệ thống** hiển thị thông báo yêu cầu "Vui lòng đăng nhập". |

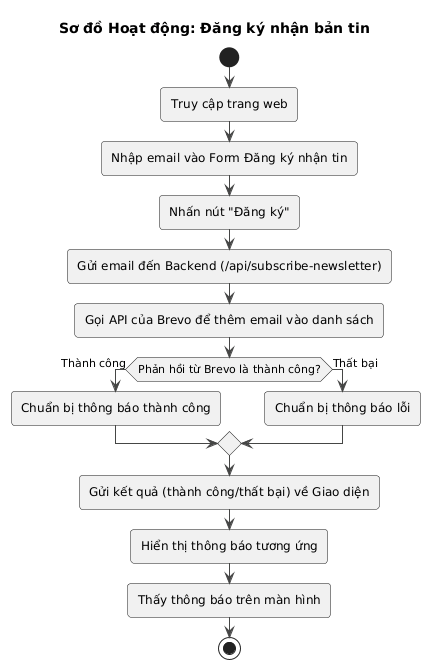
A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.7 Sơ đồ hoạt động: quản lý tài khoản cá nhân

Bảng 4.0.6 Mô tả sơ đồ hoạt động: quản lý tài khoản cá nhân

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quản lý Tài khoản Cá nhân | |
| **Tác nhân** | Khách hàng, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Xem lịch sử đơn hàng | 1. **Khách hàng** truy cập trang "Hồ sơ của tôi".  2. **Khách hàng** chọn tab "Lịch sử Đơn hàng".  3. **Hệ thống** hiển thị danh sách các đơn hàng đã đặt.  4. **Khách hàng** có thể nhấn vào một đơn hàng cụ thể để xem thông tin chi tiết. |
| Luồng chính | Quản lý sổ địa chỉ | 1. **Khách hàng** truy cập trang "Hồ sơ của tôi".  2. **Khách hàng** chọn tab "Sổ địa chỉ".  3. **Hệ thống** hiển thị các địa chỉ đã được lưu.  4. **Khách hàng** có thể thực hiện các thao tác Thêm, Sửa hoặc Xóa một địa chỉ. |
| Luồng chính | Cài đặt tài khoản | 1. **Khách hàng** truy cập trang "Hồ sơ của tôi".  2. **Khách hàng** chọn tab "Cài đặt".  3. **Hệ thống** cho phép **Khách hàng** thay đổi thông tin cá nhân như tên, ảnh đại diện (avatar).  4. **Khách hàng** có tùy chọn để xóa tài khoản vĩnh viễn nếu muốn. |



Hình 4.0.8 Sơ đồ hoạt động: Đăng ký nhận bản tin

Bảng 4.0.7 Mô tả sơ đồ hoạt động: Đăng ký nhận bản tin

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Đăng ký nhận bản tin | |
| **Tác nhân** | Người dùng, Giao diện (Frontend), Backend, Brevo API | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| **Luồng chính** | Đăng ký thành công | 1. **Người dùng** truy cập trang web, nhập email vào form và nhấn nút "Đăng ký".  2. **Giao diện** gửi thông tin email đến **Backend** thông qua một API endpoint.  3. **Backend** nhận được yêu cầu và gọi đến API của dịch vụ bên thứ ba (**Brevo**) để thêm email vào danh sách.  4. Dịch vụ **Brevo** xử lý thành công và trả về phản hồi tích cực.  5. **Backend** gửi kết quả thành công về lại cho **Giao diện**.  6. **Giao diện** hiển thị một thông báo thành công cho **Người dùng** (ví dụ: "Cảm ơn bạn đã đăng ký!"). |
| **Luồng ngoại lệ** | Đăng ký thất bại | 1. Quy trình diễn ra tương tự đến bước 3.  2. Khi **Backend** gọi API của **Brevo**, dịch vụ này trả về một lỗi (ví dụ: email không hợp lệ, email đã tồn tại, lỗi hệ thống...).  3. **Backend** nhận được phản hồi lỗi và gửi thông tin lỗi này về cho **Giao diện**.  4. **Giao diện** hiển thị một thông báo lỗi tương ứng cho **Người dùng** (ví dụ: "Email này không hợp lệ, vui lòng thử lại."). |

### 4.3.2 Sơ đồ hoạt động của quản trị viên/ nhân viên

A diagram of a workflow

AI-generated content may be incorrect.

Hình 4.0.9 Sơ đồ hoạt động: Nhập kho hàng loạt từ Excel

Bảng 4.0.8 Mô tả sơ đồ hoạt động: Nhập kho hàng loạt từ Excel

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Nhập kho Hàng loạt từ Excel | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Hoàn tất nhập kho thành công | 1. **Admin** truy cập trang "Quản lý Nhập kho" và chọn chức năng "Nhập từ Excel".  2. **Admin** chọn và tải lên một file Excel chứa danh sách sản phẩm cần nhập.  3. **Hệ thống** đọc file và xử lý thông tin của từng dòng (chi tiết ở các luồng phụ bên dưới).  4. Sau khi xử lý xong tất cả các dòng, **Hệ thống** thực hiện một giao dịch (Batch Write) để cập nhật toàn bộ dữ liệu một cách nhất quán.  5. **Hệ thống** hiển thị thông báo "Nhập kho thành công!". |
| Luồng phụ | Xử lý sản phẩm đã tồn tại | 1. Với mỗi dòng trong file Excel, **Hệ thống** đọc thông tin.  2. **Hệ thống** kiểm tra và xác định sản phẩm này đã tồn tại trong cơ sở dữ liệu.  3. **Hệ thống** tiến hành cập nhật số lượng tồn kho cho sản phẩm đó.  4. **Hệ thống** ghi nhận thông tin chi tiết vào phiếu nhập kho. |
| Xử lý sản phẩm mới | 1. Với mỗi dòng trong file Excel, **Hệ thống** đọc thông tin.  2. **Hệ thống** kiểm tra và xác định sản phẩm này chưa tồn tại.  3. **Hệ thống** tự động tạo một sản phẩm mới trong cơ sở dữ liệu dựa trên thông tin từ file.  4. **Hệ thống** cập nhật số lượng tồn kho cho sản phẩm vừa được tạo.  5. **Hệ thống** ghi nhận thông tin chi tiết vào phiếu nhập kho. |

A diagram with text and images

AI-generated content may be incorrect.

Hình 4.0.10 Sơ đồ hoạt động: Xử lý đơn hàng

Bảng 4.0.9 Mô tả sơ đồ hoạt động: Xử lý đơn hàng

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Xử lý Đơn hàng | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Xử lý thành công một đơn hàng | 1. **Admin** truy cập trang "Quản lý Đơn hàng" và xem danh sách các đơn hàng mới (trạng thái "Đang xử lý").  2. **Admin** chọn một đơn hàng, kiểm tra thông tin và tiến hành chuẩn bị hàng hóa.  3. **Admin** cập nhật trạng thái đơn hàng thành "Đang giao hàng" và bàn giao cho đơn vị vận chuyển.  4. Sau khi đơn vị vận chuyển giao hàng thành công, **Admin** cập nhật trạng thái cuối cùng thành "Hoàn thành".  5. **Hệ thống** tự động cộng điểm thưởng cho tài khoản của khách hàng. |
| Luồng phụ | Không có đơn hàng mới | 1. **Admin** truy cập trang "Quản lý Đơn hàng".  2. **Hệ thống** hiển thị danh sách và **Admin** xác định không có đơn hàng nào mới cần xử lý.  3. Quy trình kết thúc mà không có thao tác nào khác được thực hiện. |

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.11 Sơ đồ hoạt động: Quản lý & Phân quyền người dùng

Bảng 4.0.10 Mô tả sơ đồ hoạt động: Quản lý & Phân quyền người dùng

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quản lý & Phân quyền Người dùng | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Thay đổi vai trò người dùng thành công | 1. **Admin** truy cập trang "Quản lý Người dùng" và tìm đến một tài khoản cụ thể.  2. **Admin** nhấn vào danh sách vai trò, chọn một vai trò mới và xác nhận thay đổi.  3. **Hệ thống** tiến hành cập nhật lại vai trò (role) mới cho người dùng trong cơ sở dữ liệu.  4. **Hệ thống** hiển thị thông báo "Cập nhật thành công!" để xác nhận thao tác. |
| Luồng phụ | Hủy bỏ thao tác thay đổi | 1. **Admin** truy cập trang "Quản lý Người dùng" và tìm đến một tài khoản cụ thể.  2. **Admin** chọn một vai trò mới nhưng sau đó chọn "Không" ở bước xác nhận hoặc hủy bỏ thao tác.  3. **Hệ thống** không thực hiện bất kỳ thay đổi nào trong cơ sở dữ liệu và quy trình kết thúc. |

A diagram of a work flow

AI-generated content may be incorrect.

Hình 4.0.12 Sơ đồ hoạt động: Quản lý nội dung (CMS)

Bảng 4.0.11 Mô tả sơ đồ hoạt động: Quản lý nội dung (CMS)

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quản lý Nội dung (CMS) | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Thêm mới nội dung thành công | 1. **Admin** truy cập trang "Quản lý Nội dung" và chọn "Thêm mới".  2. **Hệ thống** hiển thị một form nội dung trống.  3. **Admin** điền các thông tin cần thiết (tiêu đề, nội dung, URL...) và nhấn "Lưu".  4. **Hệ thống** lưu trữ dữ liệu thành công lên cơ sở dữ liệu (Firestore).  5. **Hệ thống** hiển thị thông báo thành công và điều hướng Admin quay lại trang danh sách nội dung. |
| Luồng phụ | Chỉnh sửa nội dung thành công | 1. **Admin** truy cập trang "Quản lý Nội dung", chọn một mục đã có và nhấn "Sửa".  2. **Hệ thống** hiển thị form với các thông tin cũ của nội dung đó.  3. **Admin** thực hiện chỉnh sửa và nhấn "Lưu".  4. **Hệ thống** cập nhật dữ liệu đã thay đổi lên cơ sở dữ liệu.  5. **Hệ thống** hiển thị thông báo thành công và điều hướng Admin quay lại trang danh sách. |
| Luồng ngoại lệ | Lưu nội dung thất bại | 1. Sau khi **Admin** điền/chỉnh sửa thông tin và nhấn "Lưu".  2. **Hệ thống** xử lý và gặp phải lỗi (ví dụ: dữ liệu không hợp lệ, mất kết nối cơ sở dữ liệu).  3. **Hệ thống** hiển thị một thông báo lỗi để thông báo cho Admin.  4. Luồng kết thúc bằng việc quay lại trang danh sách nội dung. |

A diagram of a work flow

AI-generated content may be incorrect.

Hình 4.0.13 Sơ đồ hoạt động: Quản lý sản phẩm & Gán nhãn hiệu hàng loạt

Bảng 4.0.12 Mô tả sơ đồ hoạt động: Quản lý sản phẩm & Gán nhãn hiệu hàng loạt

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Quản lý Sản phẩm & Gán Nhãn hiệu Hàng loạt | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Gán nhãn hiệu hàng loạt | 1. **Admin** truy cập trang "Quản lý Sản phẩm".  2. **Admin** tick chọn nhiều sản phẩm từ danh sách.  3. **Hệ thống** hiển thị thanh công cụ cho hành động hàng loạt.  4. **Admin** chọn một nhãn hiệu từ danh sách thả xuống và xác nhận.  5. **Hệ thống** cập nhật trường nhãn hiệu (brandId) cho tất cả các sản phẩm đã chọn và hiển thị thông báo "Gán thành công!". |
| Luồng phụ | Chỉnh sửa một sản phẩm | 1. **Admin** truy cập trang "Quản lý Sản phẩm" và tìm đến một sản phẩm cụ thể.  2. **Admin** nhấn nút "Sửa" trên sản phẩm đó.  3. **Hệ thống** hiển thị một form chứa thông tin chi tiết của sản phẩm.  4. **Admin** chỉnh sửa thông tin và nhấn "Lưu".  5. **Hệ thống** cập nhật lại thông tin sản phẩm trong cơ sở dữ liệu. |
| Luồng phụ | Kết thúc (Không thao tác) | 1. **Admin** truy cập trang "Quản lý Sản phẩm".  2. **Admin** xem danh sách nhưng không thực hiện hành động gán hàng loạt hay sửa sản phẩm.  3. Quy trình kết thúc. |

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.14 Sơ đồ hoạt động: Thêm sản phẩm mới

Bảng 4.0.13 Mô tả sơ đồ hoạt động: Thêm sản phẩm mới

|  |  |  |
| --- | --- | --- |
| **Tên hoạt động** | Thêm Sản phẩm Mới | |
| **Tác nhân** | Admin, Hệ thống | |
| **Loại luồng** | **Tên luồng** | **Mô tả chi tiết các bước** |
| Luồng chính | Thêm sản phẩm mới thành công | 1. **Admin** truy cập trang "Quản lý Sản phẩm" và nhấn nút "Thêm sản phẩm mới".  2. **Hệ thống** hiển thị form nhập liệu.  3. **Admin** điền đầy đủ và hợp lệ các thông tin được yêu cầu (tên, giá, danh mục, tồn kho, hình ảnh...).  4. **Admin** nhấn "Lưu sản phẩm".  5. **Hệ thống** xác thực dữ liệu thành công, thực hiện một giao dịch (Batch Write) để tạo sản phẩm và các thông tin tồn kho liên quan.  6. **Hệ thống** hiển thị thông báo "Lưu thành công!", sau đó đóng form và quay lại trang danh sách sản phẩm. |
| Luồng ngoại lệ | Dữ liệu nhập không hợp lệ | 1. **Admin** điền thiếu hoặc sai định dạng thông tin trên form và nhấn "Lưu sản phẩm".  2. **Hệ thống** kiểm tra và phát hiện dữ liệu không hợp lệ.  3. **Hệ thống** hiển thị một thông báo lỗi chi tiết để hướng dẫn Admin.  4. **Hệ thống** giữ nguyên trạng thái form và các dữ liệu đã nhập để **Admin** có thể dễ dàng chỉnh sửa lại. |

## 4.4 Sơ đồ tuần tự

### 4.4.1 Sơ đồ tuần tự của khách hàng

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 4.0.15 Sơ đồ tuần tự: Cập nhật thông cá nhân

Bảng 4.0.14 Mô tả sơ đồ tuần tự: Cập nhật thông cá nhân

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên quy trình** | | Cập nhật Thông tin Cá nhân (Tên hiển thị) | |
| **Các bên tham gia** | | Khách hàng, Giao diện (React), Xác thực (Firebase Auth), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Cập nhật Tên hiển thị thành công | | 1. **Khách hàng** vào trang "Hồ sơ của tôi" và nhấn "Chỉnh sửa" trên **Giao diện**.  2. **Giao diện** hiển thị form, **Khách hàng** thay đổi Tên hiển thị và nhấn "Lưu".  3. **Giao diện** gọi hàm xử lý handleSaveProfile() để bắt đầu quá trình cập nhật.  4. **Giao diện** gửi yêu cầu updateProfile() đến dịch vụ **Xác thực (Firebase Auth)** để cập nhật thông tin định danh.  5. **Giao diện** tiếp tục gửi yêu cầu updateDoc() đến **Cơ sở dữ liệu (Firestore)** để cập nhật thông tin trong document của người dùng.  6. Sau khi cả hai dịch vụ backend xác nhận thành công, **Giao diện** hiển thị thông báo "Cập nhật thành công!" cho **Khách hàng**.  7. **Giao diện** tự động chuyển về chế độ xem thông tin. |

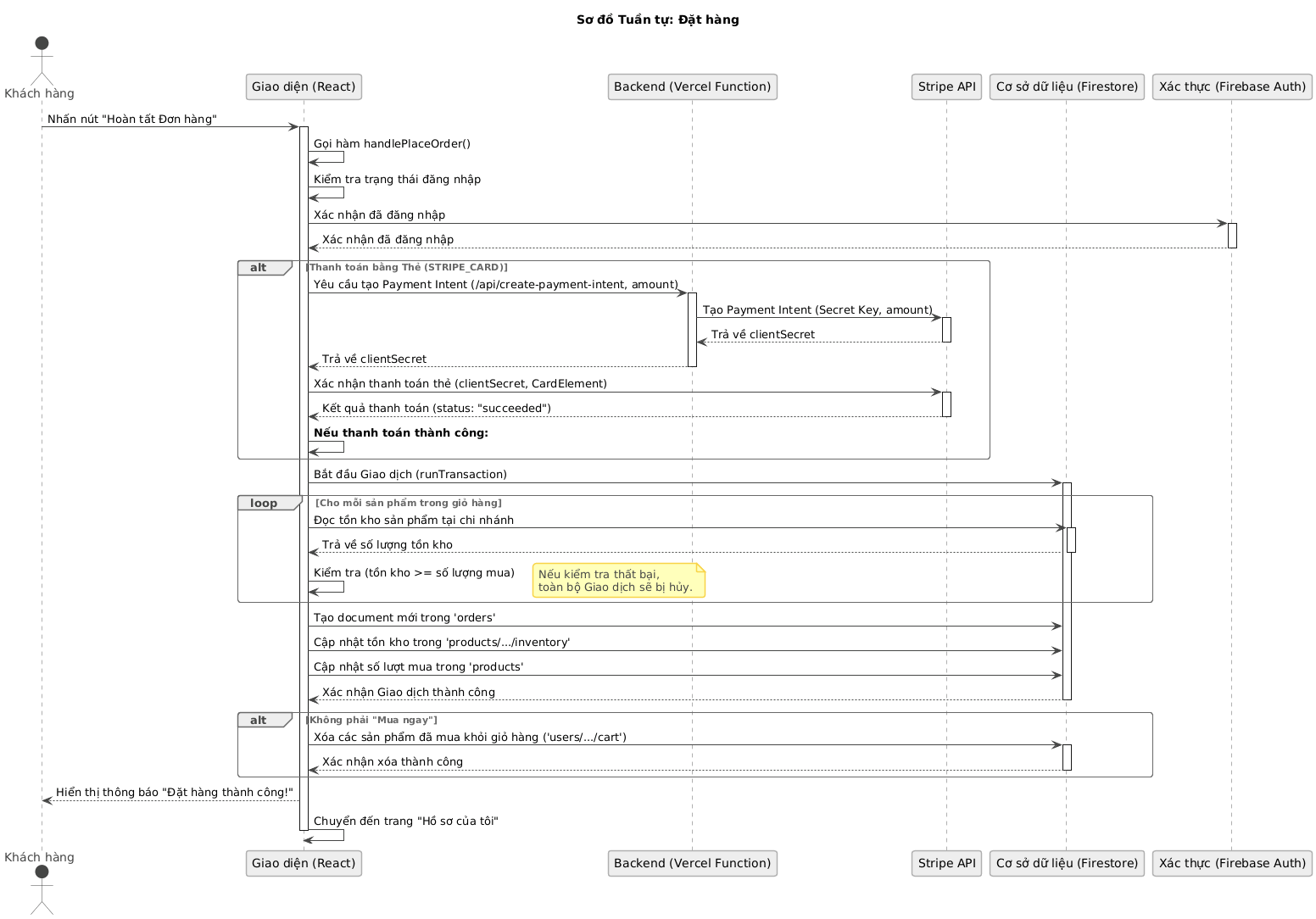
A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 4.0.16 Sơ đồ tuần tự: Đăng nhập người dùng

Bảng 4.0.15 Mô tả sơ đồ tuần tự: Đăng nhập người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên quy trình** | | Đăng nhập Người dùng | |
| **Các bên tham gia** | | Khách hàng, Giao diện (React), Xác thực (Firebase Auth), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Đăng nhập thành công | | 1. **Khách hàng** nhập Email, Mật khẩu và nhấn "Đăng nhập" trên **Giao diện**.  2. **Giao diện** gửi yêu cầu signInWithEmailAndPassword() đến dịch vụ **Xác thực (Firebase Auth)**.  3. **Xác thực** kiểm tra thông tin, nếu hợp lệ, sẽ trả về thông tin định danh (User Credential) cho **Giao diện**.  4. Sự kiện đăng nhập thành công này kích hoạt một bộ lắng nghe (onAuthStateChanged listener) trên **Giao diện**.  5. Bên trong listener, **Giao diện** dùng UID (User ID) của người dùng để gửi yêu cầu lấy dữ liệu chi tiết từ **Cơ sở dữ liệu (Firestore)**.  6. **Cơ sở dữ liệu** trả về document chứa thông tin của người dùng.  7. **Giao diện** cập nhật trạng thái toàn cục (global state) với đầy đủ thông tin người dùng vừa nhận được.  8. Cuối cùng, **Giao diện** đóng Modal đăng nhập và cập nhật Header để hiển thị tên người dùng, báo hiệu đăng nhập thành công. |



Hình 4.0.17 Sơ đồ tuần tự: Đặt hàng

Bảng 4.0.16 Mô tả sơ đồ tuần tự: Đặt hàng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên quy trình** | | Đặt hàng (Place Order) | |
| **Các bên tham gia** | | Khách hàng, Giao diện (React), Backend (Vercel Function), Stripe API, Cơ sở dữ liệu (Firestore), Xác thực (Firebase Auth) | |
| **Loại kịch bản** | **Tên kịch bản** | | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Đặt hàng thành công (Thanh toán bằng thẻ) | | 1. **Khách hàng** nhấn "Hoàn tất Đơn hàng". **Giao diện** kiểm tra trạng thái đăng nhập với **Xác thực (Firebase Auth)**.  2. **Giao diện** yêu cầu **Backend** tạo một phiên thanh toán (Payment Intent).  3. **Backend** gọi đến **Stripe API** để tạo Payment Intent và nhận về một mã bí mật (clientSecret).  4. **Giao diện** sử dụng clientSecret này để gửi thông tin thẻ và xác nhận thanh toán với **Stripe API**.  5. Khi thanh toán thẻ thành công, **Giao diện** yêu cầu **Backend** bắt đầu một Giao dịch (Transaction) để xử lý logic đơn hàng.  6. **Backend** thực hiện vòng lặp kiểm tra tồn kho, sau đó ghi một loạt dữ liệu (tạo đơn hàng mới, cập nhật tồn kho, cập nhật số lượt mua...) vào **Cơ sở dữ liệu (Firestore)** trong cùng một giao dịch.  7. (Tùy chọn) **Backend** tiếp tục xóa các sản phẩm đã mua khỏi giỏ hàng của người dùng.  8. Cuối cùng, **Giao diện** hiển thị thông báo "Đặt hàng thành công!" cho **Khách hàng**. |
| Kịch bản phụ | Đặt hàng (Không dùng thẻ) | | 1. Trong trường hợp người dùng chọn một phương thức khác (ví dụ: Thanh toán khi nhận hàng - COD), quy trình sẽ bỏ qua toàn bộ khối alt tương tác với **Stripe API** (bước 2, 3, 4).  2. **Giao diện** sẽ trực tiếp yêu cầu **Backend** bắt đầu Giao dịch để xử lý đơn hàng như ở bước 5 của kịch bản chính. |
| Kịch bản ngoại lệ | Đặt hàng thất bại (Không đủ tồn kho) | | 1. Quy trình diễn ra tương tự kịch bản chính cho đến khi **Backend** bắt đầu Giao dịch.  2. Trong vòng lặp kiểm tra tồn kho, **Backend** phát hiện một sản phẩm trong giỏ không đủ số lượng.  3. Toàn bộ Giao dịch với **Cơ sở dữ liệu (Firestore)** sẽ tự động bị hủy. Sẽ không có đơn hàng nào được tạo và tồn kho không bị thay đổi.  4. **Giao diện** sẽ nhận được phản hồi lỗi từ **Backend** và thông báo cho **Khách hàng**. |

A close-up of a diagram

AI-generated content may be incorrect.

Hình 4.0.18 Sơ đồ tuần tự: Đổi điểm thưởng lấy Voucher

Bảng 4.0.17 Mô tả sơ đồ tuần tự: Đổi điểm thưởng lấy Voucher

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Đổi Điểm thưởng lấy Voucher | |
| **Các bên tham gia** | Khách hàng, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Đổi điểm thành công | 1. **Khách hàng** vào trang "Điểm thưởng" và nhấn nút "Đổi" trên **Giao diện**.  2. **Giao diện** gọi hàm xử lý, kiểm tra và xác nhận khách hàng có đủ điểm để thực hiện đổi thưởng.  3. **Giao diện** bắt đầu một Giao dịch ghi hàng loạt (writeBatch) đến **Cơ sở dữ liệu (Firestore)** để đảm bảo tính toàn vẹn dữ liệu.  4. Trong cùng giao dịch, **Giao diện** gửi 3 lệnh ghi: trừ điểm trong tài khoản người dùng, tạo một bản ghi lịch sử đổi điểm, và tạo một mã voucher mới trong danh sách khuyến mãi.  5. **Cơ sở dữ liệu** xác nhận toàn bộ giao dịch được thực hiện thành công.  6. **Giao diện** hiển thị thông báo "Đổi điểm thành công!" kèm theo mã voucher vừa được tạo cho **Khách hàng**. |
| Kịch bản ngoại lệ | Đổi điểm thất bại (Không đủ điểm) | 1. **Khách hàng** vào trang "Điểm thưởng" và nhấn nút "Đổi".  2. **Giao diện** gọi hàm xử lý và thực hiện kiểm tra (được ghi chú trong ref trên sơ đồ).  3. **Hệ thống** phát hiện số điểm của khách hàng không đủ để thực hiện quy đổi.  4. Quy trình sẽ dừng lại và **Giao diện** sẽ hiển thị thông báo lỗi cho **Khách hàng** (ví dụ: "Bạn không đủ điểm"). Không có giao dịch nào được gửi đến cơ sở dữ liệu. |

A screenshot of a diagram

AI-generated content may be incorrect.

Hình 4.0.19 Sơ tuần tự: Gửi đánh giá đơn hàng

Bảng 4.0.18 Mô tả sơ tuần tự: Gửi đánh giá đơn hàng

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Gửi Đánh giá Đơn hàng | |
| **Các bên tham gia** | Khách hàng, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Gửi đánh giá thành công | 1. **Khách hàng** truy cập trang "Đánh giá Đơn hàng".  2. **Giao diện** hiển thị danh sách các sản phẩm trong đơn hàng đó.  3. **Khách hàng** chọn sao (rating), viết bình luận cho các sản phẩm và nhấn nút "Hoàn tất đánh giá".  4. **Giao diện** bắt đầu một Giao dịch (Transaction) với **Cơ sở dữ liệu (Firestore)** để đảm bảo tính nhất quán.  5. **Vòng lặp (Loop):** Với mỗi sản phẩm được đánh giá, **Giao diện** đọc dữ liệu xếp hạng hiện tại, tính toán lại điểm trung bình mới, sau đó gửi lệnh cập nhật điểm trung bình và tạo một document review mới cho sản phẩm đó.  6. Sau khi vòng lặp kết thúc, **Giao diện** gửi thêm một lệnh cập nhật trạng thái của đơn hàng thành "đã đánh giá" (reviewed: true).  7. Toàn bộ các lệnh ghi trong giao dịch được **Cơ sở dữ liệu** xác nhận thành công.  8. **Giao diện** hiển thị thông báo "Cảm ơn bạn đã đánh giá!" và chuyển hướng người dùng về trang cá nhân. |
| Kịch bản ngoại lệ | Gửi đánh giá thất bại (Lỗi giao dịch) | 1. Quy trình diễn ra tương tự đến khi bắt đầu Giao dịch.  2. Trong quá trình thực hiện các lệnh đọc/ghi bên trong vòng lặp, có một lỗi xảy ra (ví dụ: một sản phẩm bị xóa bởi Admin cùng lúc).  3. Toàn bộ Giao dịch sẽ bị hủy. Điều này đảm bảo không có đánh giá nào được lưu và điểm trung bình của các sản phẩm khác không bị cập nhật sai lệch.  4. **Giao diện** sẽ hiển thị một thông báo lỗi cho **Khách hàng**. |

A screenshot of a chat

AI-generated content may be incorrect.

Hình 4.0.20 Sơ đồ tuần tự: Thêm sản phẩm vào giỏ hàng

Bảng 4.0.19 Mô tả sơ đồ tuần tự: Thêm sản phẩm vào giỏ hàng

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Thêm sản phẩm vào Giỏ hàng | |
| **Các bên tham gia** | Khách hàng, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Thêm/Cập nhật sản phẩm trong giỏ hàng thành công | 1. **Khách hàng** nhấn nút "Thêm vào giỏ" trên một sản phẩm.  2. **Giao diện** gọi hàm addToCart() và gửi một yêu cầu ghi (setDoc) đến **Cơ sở dữ liệu (Firestore)**. Yêu cầu này được thiết lập để:  • Tạo mới sản phẩm trong giỏ nếu chưa có.  • Hoặc tăng số lượng lên 1 nếu sản phẩm đã có sẵn (sử dụng increment).  3. **Cơ sở dữ liệu** xác nhận thao tác ghi thành công, và **Giao diện** ngay lập tức hiển thị thông báo "Thêm thành công!" cho khách hàng.  4. **Quan trọng:** Một bộ lắng nghe thời gian thực (onSnapshot) đã được kích hoạt trên **Giao diện** sẽ tự động phát hiện sự thay đổi trong giỏ hàng trên **Cơ sở dữ liệu**.  5. **Cơ sở dữ liệu** gửi lại toàn bộ dữ liệu giỏ hàng đã được cập nhật cho **Giao diện**.  6. **Giao diện** dùng dữ liệu này để cập nhật lại trạng thái (state) giỏ hàng của ứng dụng.  7. Cuối cùng, **Giao diện** cập nhật lại icon giỏ hàng trên Header (ví dụ: hiển thị đúng tổng số lượng sản phẩm). |

A screenshot of a diagram

AI-generated content may be incorrect.

Hình 4.0.21 Sơ đồ tuần tự: Tìm kiếm và Xem chi tiết sản phẩm

Bảng 4.0.20 Mô tả sơ đồ tuần tự: Tìm kiếm và Xem chi tiết sản phẩm

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Tìm kiếm và Xem chi tiết Sản phẩm | |
| **Các bên tham gia** | Khách hàng, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Tìm kiếm và xem sản phẩm thành công | Quy trình gồm 2 giai đoạn chính:   * + 1. **Tìm kiếm sản phẩm:**   • **Khách hàng** nhập từ khóa (ví dụ: "cà chua") vào thanh tìm kiếm và nhấn Enter.  • **Giao diện** gửi một truy vấn đến **Cơ sở dữ liệu (Firestore)** để lấy về danh sách sản phẩm.  • Sau khi nhận được danh sách, **Giao diện** tự thực hiện việc lọc ra các sản phẩm khớp với từ khóa của người dùng.  • **Giao diện** hiển thị kết quả là một lưới các sản phẩm đã lọc cho **Khách hàng** xem.  **2. Xem chi tiết sản phẩm:**  • **Khách hàng** nhấn vào một sản phẩm từ kết quả tìm kiếm.  • **Giao diện** điều hướng sang trang chi tiết sản phẩm (Product Detail Page).  • Tại đây, **Giao diện** gửi các yêu cầu mới đến **Cơ sở dữ liệu** để lấy về dữ liệu chi tiết và danh sách các đánh giá liên quan đến sản phẩm đó.  • Sau khi nhận đủ dữ liệu, **Giao diện** hiển thị trang chi tiết sản phẩm đầy đủ cho **Khách hàng**. |

A diagram of a diagram

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Tải trang Chi tiết Sản phẩm và Cập nhật SEO | |
| **Các bên tham gia** | Khách hàng, Trình duyệt, Giao diện (React), CSDL (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| **Kịch bản chính** | Tải trang và cập nhật SEO thành công | 1. **Khách hàng** truy cập một URL sản phẩm cụ thể trên **Trình duyệt**.  2. **Trình duyệt** tải về các file HTML và Javascript ban đầu của ứng dụng React.  3. **Giao diện (React)** được khởi chạy và render component trang chi tiết sản phẩm (ProductDetailPage).  4. Component này ngay lập tức gửi yêu cầu đến **CSDL (Firestore)** để lấy dữ liệu chi tiết của sản phẩm tương ứng với URL.  5. Sau khi nhận được dữ liệu (tên, mô tả, ảnh...), **Giao diện** render một component <SEO> đặc biệt (sử dụng thư viện React Helmet Async).  6. Component <SEO> này có nhiệm vụ dùng dữ liệu sản phẩm vừa nhận được để cập nhật lại các thẻ <title> và <meta> trong phần <head> của trang web.  7. **Trình duyệt** nhận được sự thay đổi này và **Khách hàng** sẽ thấy trang web hiển thị hoàn chỉnh với tiêu đề và các thẻ meta chính xác, giúp các công cụ tìm kiếm (Google) đọc được nội dung trang. |

### 4.4.2 Sơ đồ tuần tự của quản trị viên

A screenshot of a chat

AI-generated content may be incorrect.

Hình 4.0.22 Sơ đồ tuần tự: Admin cập nhật trạng thái đơn hàng

Bảng 4.0.21 Mô tả sơ đồ tuần tự: Admin cập nhật trạng thái đơn hàng

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Cập nhật Trạng thái Đơn hàng | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| **Kịch bản chính** | **Cập nhật trạng thái thành công** | 1. **Admin** truy cập trang "Quản lý Đơn hàng".  2. **Giao diện** gửi yêu cầu lấy danh sách đơn hàng từ **Cơ sở dữ liệu (Firestore)** và hiển thị lên cho **Admin** xem.  3. **Admin** chọn một trạng thái mới (ví dụ: "Đang giao hàng") cho một đơn hàng cụ thể từ danh sách.  4. **Giao diện** gọi hàm xử lý và gửi yêu cầu cập nhật (updateDoc) trạng thái mới cho đơn hàng đó lên **Cơ sở dữ liệu**.  5. **Quan trọng:** Một bộ lắng nghe thời gian thực (onSnapshot) đang theo dõi toàn bộ danh sách đơn hàng sẽ tự động được kích hoạt bởi sự thay đổi ở bước 4.  6. **Cơ sở dữ liệu** gửi lại toàn bộ danh sách đơn hàng đã được làm mới cho **Giao diện**.  7. **Giao diện** cập nhật lại trạng thái (state) của mình và tự động render lại bảng danh sách.  8. Kết quả là **Admin** sẽ nhìn thấy trạng thái mới của đơn hàng được cập nhật trực tiếp trên bảng một cách tự động. |

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 4.0.23 Sơ đồ tuần tự: Admin gán nhãn hiệu hàng loạt

Bảng 4.0.22 Mô tả sơ đồ tuần tự: Admin gán nhãn hiệu hàng loạt

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Gán Nhãn hiệu Hàng loạt | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Gán nhãn hiệu thành công | 1. **Admin** tick chọn nhiều sản phẩm trong bảng danh sách trên **Giao diện**.  2. **Giao diện** cập nhật trạng thái (state) nội bộ để lưu các sản phẩm đã chọn và hiển thị thanh hành động hàng loạt.  3. **Admin** chọn một nhãn hiệu từ dropdown "Gán nhãn hiệu".  4. **Giao diện** bắt đầu một Giao dịch ghi hàng loạt (writeBatch) đến **Cơ sở dữ liệu (Firestore)**.  5. **Vòng lặp (Loop):** Với mỗi sản phẩm đã được **Admin** chọn, **Giao diện** thêm một lệnh cập nhật trường brandId cho sản phẩm đó vào trong batch.  6. Sau khi thêm hết các lệnh, toàn bộ batch được gửi đi và **Cơ sở dữ liệu** thực thi, sau đó xác nhận giao dịch thành công.  7. **Giao diện** reset lại danh sách sản phẩm đã chọn, hiển thị thông báo "Gán nhãn hiệu thành công!" và ẩn thanh hành động hàng loạt. |

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 4.0.24 Sơ đồ tuần tự: Admin nhập kho hàng loạt từ Excel

Bảng 4.0.23 Mô tả sơ đồ tuần tự: Admin nhập kho hàng loạt từ Excel

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Nhập kho Hàng loạt từ Excel | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Nhập kho thành công | 1. **Admin** truy cập trang "Quản lý Nhập kho", chọn một file Excel và nhấn nút "Bắt đầu nhập kho".  2. **Giao diện** gọi hàm xử lý, tiến hành đọc và phân tích dữ liệu từ file Excel đã tải lên.  3. **Giao diện** bắt đầu một Giao dịch ghi hàng loạt (writeBatch) đến **Cơ sở dữ liệu (Firestore)** để đảm bảo tất cả các cập nhật được thực hiện cùng lúc.  4. **Vòng lặp (Loop) ngoài:** Với mỗi dòng (sản phẩm) trong file Excel, **Giao diện** sẽ tìm kiếm thông tin sản phẩm tương ứng trên **Cơ sở dữ liệu**.  5. **Vòng lặp (Loop) trong:** Với mỗi chi nhánh có thông tin tồn kho cho sản phẩm đó, **Giao diện** sẽ thêm một lệnh cập nhật tồn kho vào trong batch.  6. Sau khi các vòng lặp kết thúc, **Giao diện** thêm một lệnh cuối cùng vào batch để tạo một phiếu nhập kho mới.  7. Toàn bộ batch được gửi đi và **Cơ sở dữ liệu** xác nhận giao dịch thành công.  8. **Giao diện** hiển thị thông báo "Nhập kho thành công!" cho **Admin** và đóng Modal lại. |

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 4.0.25 Sơ đồ tuần tự: Admin phản hồi bình luận

Bảng 4.0.24 Mô tả sơ đồ tuần tự: Admin phản hồi bình luận

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Phản hồi Bình luận | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Gửi phản hồi thành công | 1. **Admin** truy cập trang "Quản lý Bình luận".  2. **Giao diện** gửi một truy vấn gom nhóm (collectionGroup) đến **Cơ sở dữ liệu (Firestore)** để lấy về tất cả bình luận từ tất cả các sản phẩm và hiển thị lên bảng.  3. **Admin** chọn "Phản hồi" một bình luận cụ thể, nhập nội dung trả lời và nhấn nút "Gửi phản hồi".  4. **Giao diện** gọi hàm xử lý và gửi yêu cầu cập nhật (updateDoc) đến đúng document của bình luận đó trên **Cơ sở dữ liệu**.  5. Yêu cầu này sẽ thêm các trường thông tin về phản hồi của admin như adminReply (nội dung), repliedAt (thời gian), và repliedBy (người phản hồi).  6. **Cơ sở dữ liệu** xác nhận cập nhật thành công.  7. **Giao diện** hiển thị thông báo "Gửi phản hồi thành công!" cho **Admin** và đóng modal phản hồi lại. |

A diagram of a project

AI-generated content may be incorrect.

Hình 4.0.26 Sơ đồ tuần tự: Admin cập nhật vai trò người dùng

Bảng 4.0.25 Mô tả sơ đồ tuần tự: Admin cập nhật vai trò người dùng

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Cập nhật Vai trò Người dùng | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Cập nhật vai trò thành công | 1. **Admin** truy cập trang "Quản lý Người dùng".  2. **Giao diện** gửi yêu cầu lấy danh sách người dùng từ **Cơ sở dữ liệu (Firestore)** và hiển thị lên bảng cho **Admin** xem.  3. **Admin** chọn một vai trò mới (ví dụ: "staff") cho một người dùng cụ thể ngay trên bảng danh sách.  4. **Giao diện** gọi hàm xử lý và gửi yêu cầu cập nhật (updateDoc) trường role cho document của người dùng đó lên **Cơ sở dữ liệu**.  5. **Cơ sở dữ liệu** xác nhận thao tác cập nhật thành công.  6. **Giao diện** hiển thị thông báo "Cập nhật vai trò thành công!" cho **Admin**.  7. Đồng thời, **Giao diện** sẽ tự động cập nhật lại vai trò mới của người dùng đó ngay trên bảng (thông qua việc làm mới state hoặc listener). |

A diagram with text and words

AI-generated content may be incorrect.

Hình 4.0.27 Sơ đồ tuần tự: Admin quản lý nội dung (Thêm/Sửa)

Bảng 4.0.26 Mô tả sơ đồ tuần tự: Admin quản lý nội dung (Thêm/Sửa)

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Quản lý Nội dung (Thêm/Sửa) | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Lưu nội dung (Thêm mới hoặc Cập nhật) thành công | 1. **Admin** truy cập trang "Quản lý Nội dung".  2. **Giao diện** lấy và hiển thị danh sách các nội dung hiện có từ **Cơ sở dữ liệu (Firestore)**.  3. **Admin** chọn "Thêm mới" (để tạo nội dung mới) hoặc "Sửa" (để chỉnh sửa một nội dung có sẵn).  4. **Giao diện** hiển thị một Form để **Admin** điền hoặc chỉnh sửa các thông tin như Tiêu đề, URL (slug), Nội dung... rồi nhấn "Lưu".  5. **Giao diện** gửi một yêu cầu ghi (setDoc) đến **Cơ sở dữ liệu**.  6. **Quan trọng:** Yêu cầu này sử dụng tùy chọn {merge: true}, cho phép nó tự động **tạo mới** document nếu chưa tồn tại, hoặc **cập nhật** lên document đã có sẵn mà không ghi đè toàn bộ.  7. **Cơ sở dữ liệu** xác nhận thao tác ghi thành công.  8. **Giao diện** hiển thị thông báo "Lưu thành công!" cho **Admin**, sau đó đóng Form và quay lại trang danh sách nội dung. |

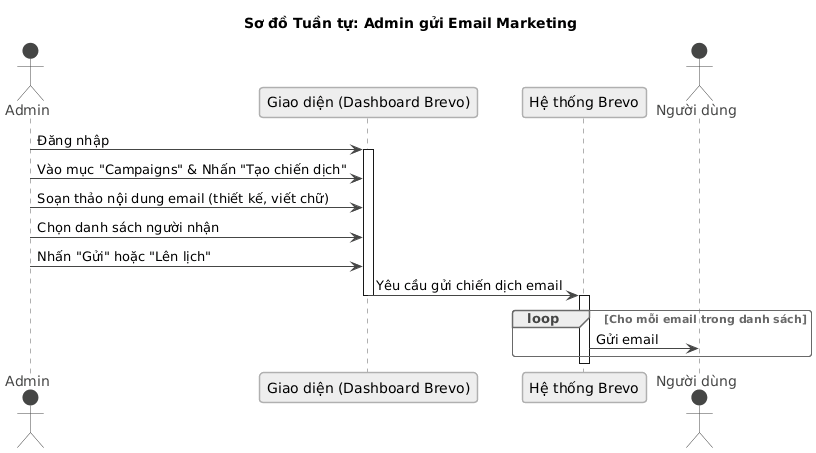
A diagram of a product

AI-generated content may be incorrect.

Hình 4.0.28 Sơ đồ tuần tự: Admin thêm sản phẩm mới

Bảng 4.0.27 Mô tả sơ đồ tuần tự: Admin thêm sản phẩm mới

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Thêm Sản phẩm Mới | |
| **Các bên tham gia** | Admin, Giao diện (React), Cơ sở dữ liệu (Firestore) | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Thêm sản phẩm và tồn kho thành công | 1. **Admin** nhấn nút "Thêm sản phẩm mới" trên **Giao diện**.  2. **Giao diện** hiển thị một form (ProductForm) để **Admin** điền thông tin sản phẩm và số lượng tồn kho cho từng chi nhánh.  3. Sau khi **Admin** nhấn "Lưu sản phẩm", **Giao diện** tự kiểm tra tính hợp lệ của dữ liệu đã nhập.  4. **Giao diện** bắt đầu một Giao dịch ghi hàng loạt (writeBatch) đến **Cơ sở dữ liệu (Firestore)** để đảm bảo tính toàn vẹn dữ liệu.  5. **Giao diện** thêm một lệnh để tạo document sản phẩm chính vào trong batch.  6. **Vòng lặp (Loop):** Với mỗi chi nhánh có nhập tồn kho, **Giao diện** thêm một lệnh tương ứng để tạo document tồn kho trong subcollection inventory của sản phẩm đó.  7. Toàn bộ batch được gửi đi và **Cơ sở dữ liệu** xác nhận giao dịch thành công.  8. **Giao diện** hiển thị thông báo "Lưu thành công!" cho **Admin**, sau đó đóng Form và quay về trang danh sách sản phẩm. |



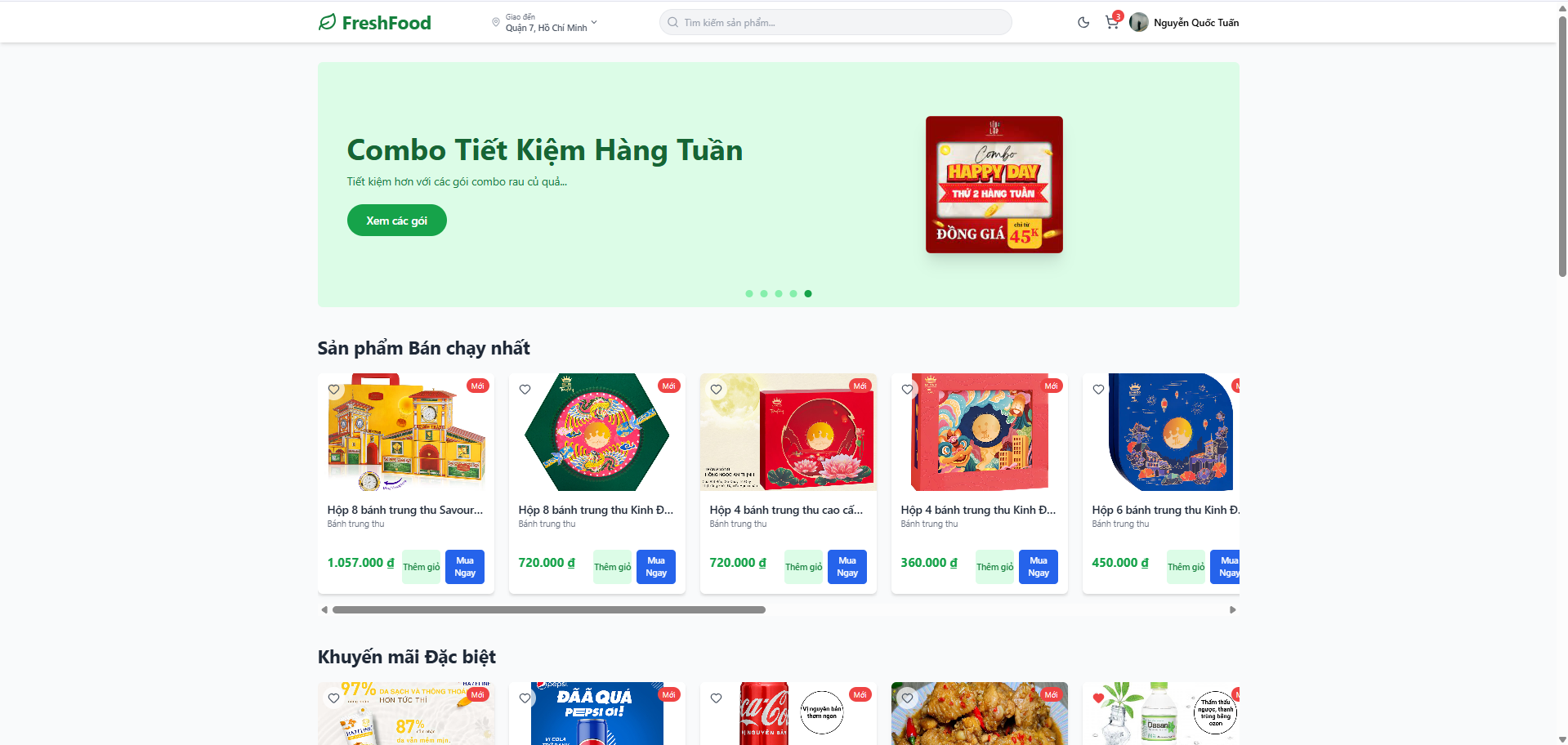
Hình 4.0.29 Sơ đồ tuần tự: Admin gửi Email Marketing qua Brevo

Bảng 4.0.28 Mô tả sơ đồ tuần tự: Admin gửi Email Marketing qua Brevo

|  |  |  |
| --- | --- | --- |
| **Tên quy trình** | Admin Gửi Email Marketing qua Brevo | |
| **Các bên tham gia** | Admin, Giao diện (Dashboard Brevo), Hệ thống Brevo, Người dùng | |
| **Loại kịch bản** | **Tên kịch bản** | **Mô tả chi tiết các bước tương tác** |
| Kịch bản chính | Gửi chiến dịch email thành công | 1. **Admin** đăng nhập vào **Giao diện (Dashboard Brevo)**.  2. **Admin** thực hiện một loạt các thao tác trên giao diện để chuẩn bị chiến dịch: vào mục "Campaigns", tạo chiến dịch mới, soạn thảo nội dung email (thiết kế, văn bản), và chọn danh sách người sẽ nhận email.  3. **Admin** nhấn nút "Gửi" hoặc "Lên lịch" để hoàn tất thiết lập.  4. **Giao diện** gửi một yêu cầu thực thi chiến dịch email đến **Hệ thống Brevo**.  5. **Hệ thống Brevo** tiếp nhận yêu cầu và bắt đầu quá trình xử lý.  6. **Vòng lặp (Loop):** Với mỗi địa chỉ email có trong danh sách người nhận đã chọn, **Hệ thống Brevo** sẽ gửi nội dung email đã được soạn đến **Người dùng** cuối. |

## 4.5. Module Giao diện Khách hàng

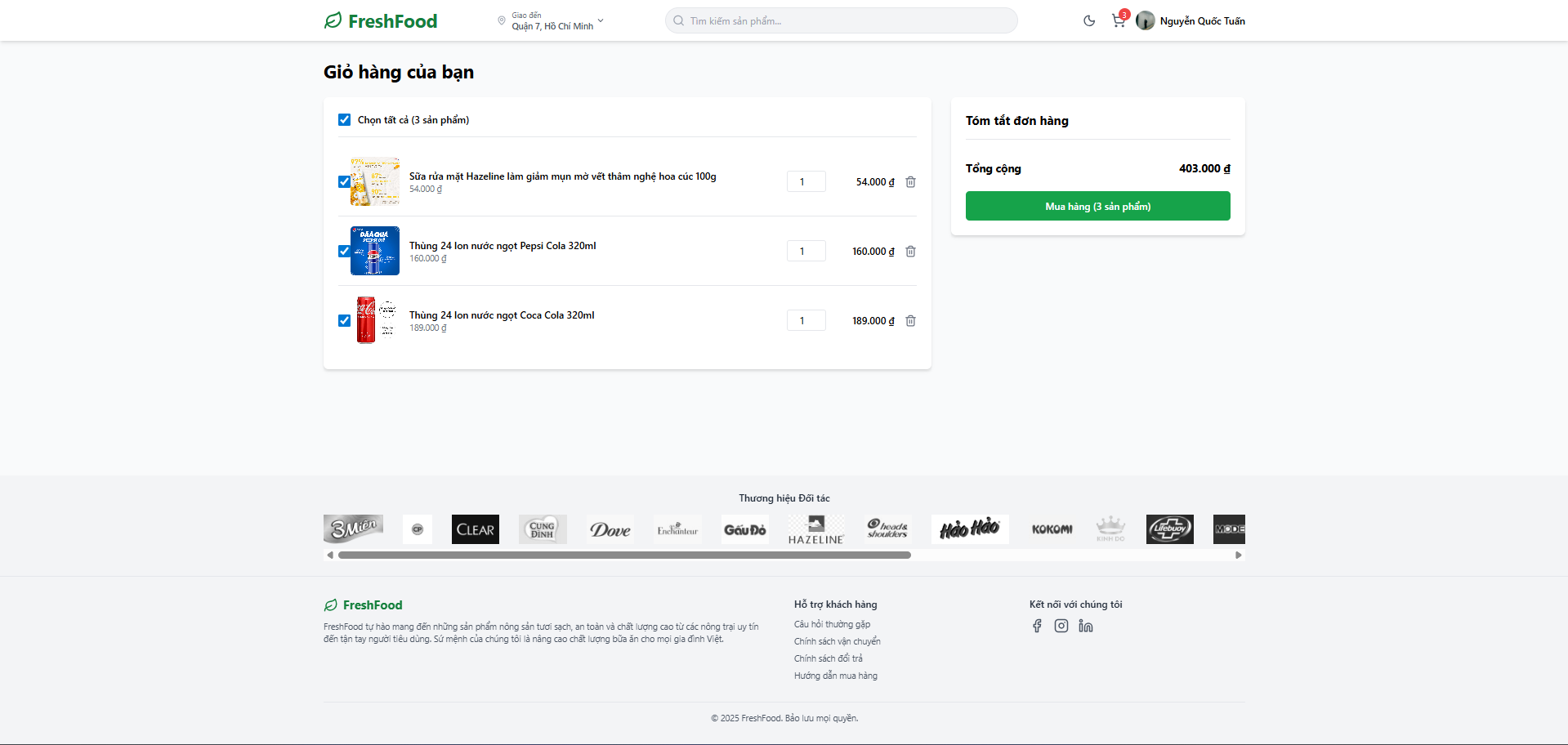
### 4.5.1. Trang chủ (HomePage) và Hiển thị Sản phẩm



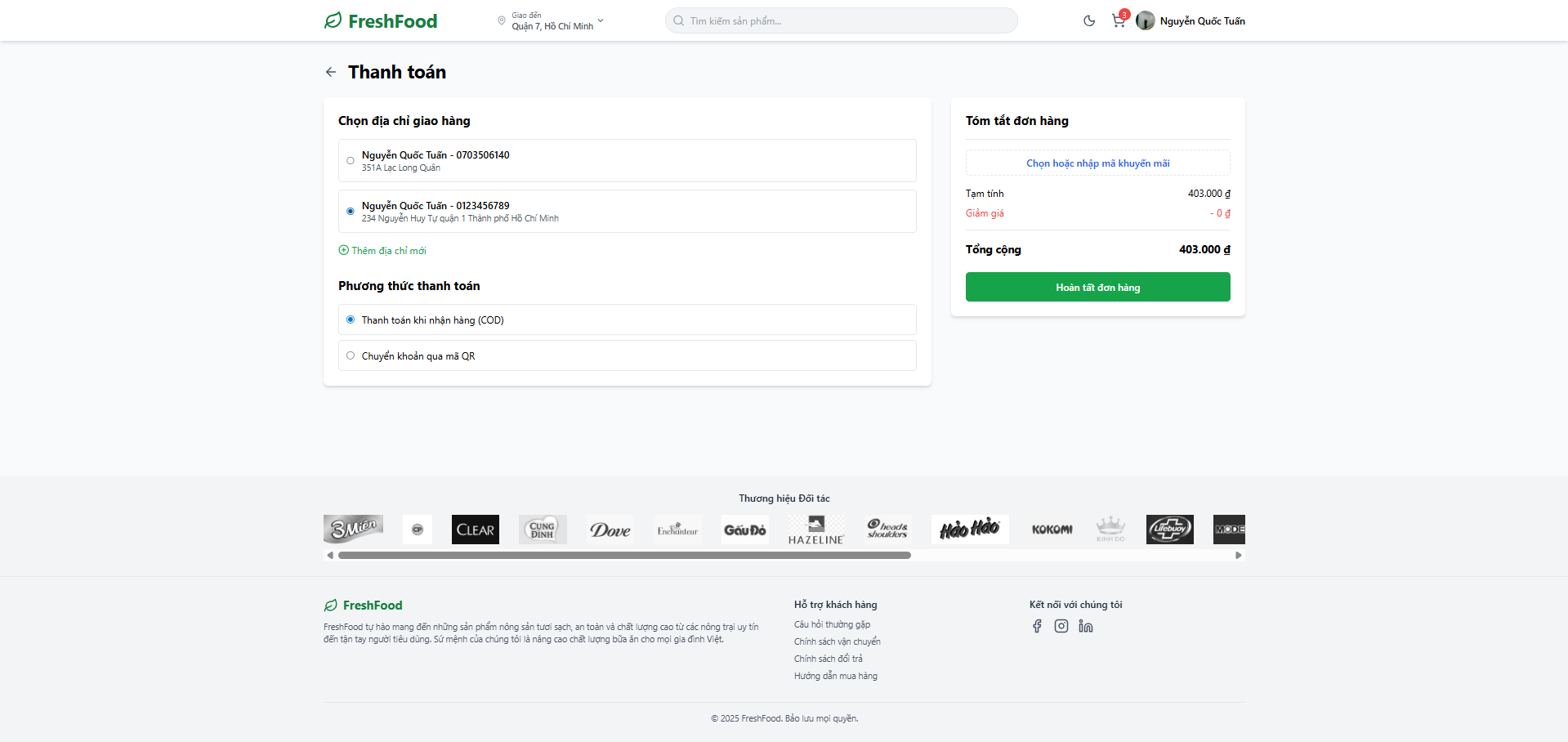
Hình 4.0.30 Trang chủ (HomePage) và Hiển thị Sản phẩm

* **Chức năng:** Đây là trang chính, nơi hiển thị các sản phẩm nổi bật, banner quảng cáo, và lưới sản phẩm chính.
* **Phân tích Code:**
  + **Tải dữ liệu:** HomePage sử dụng useEffect để thực hiện nhiều truy vấn song song đến Firestore, lấy về danh sách sản phẩm bán chạy (bestSellers), sản phẩm khuyến mãi (saleProducts), và danh sách sản phẩm chính.
  + **Lọc và Sắp xếp:** Người dùng có thể lọc sản phẩm theo danh mục (cha và con) và sắp xếp theo các tiêu chí như "Mới nhất", "Bán chạy", "Giá tăng/giảm dần". Logic lọc được thực hiện ở phía client, kết hợp các điều kiện từ các state selectedCategory, sortOption và searchQuery.
  + **Logic Đa chi nhánh:** useEffect chính được nâng cấp để chỉ hiển thị các sản phẩm có tồn kho (stock > 0) tại chi nhánh (selectedBranch) mà người dùng đã chọn, đảm bảo tính chính xác của thông tin.
  + **Component ProductCard:** Là thành phần tái sử dụng để hiển thị thông tin tóm tắt của một sản phẩm, bao gồm ảnh đại diện (thumbnailUrl), tên, giá, và các nút "Thêm vào giỏ", "Mua Ngay".

### 4.5.2. Giỏ hàng (CartPage) và Thanh toán (CheckoutPage)



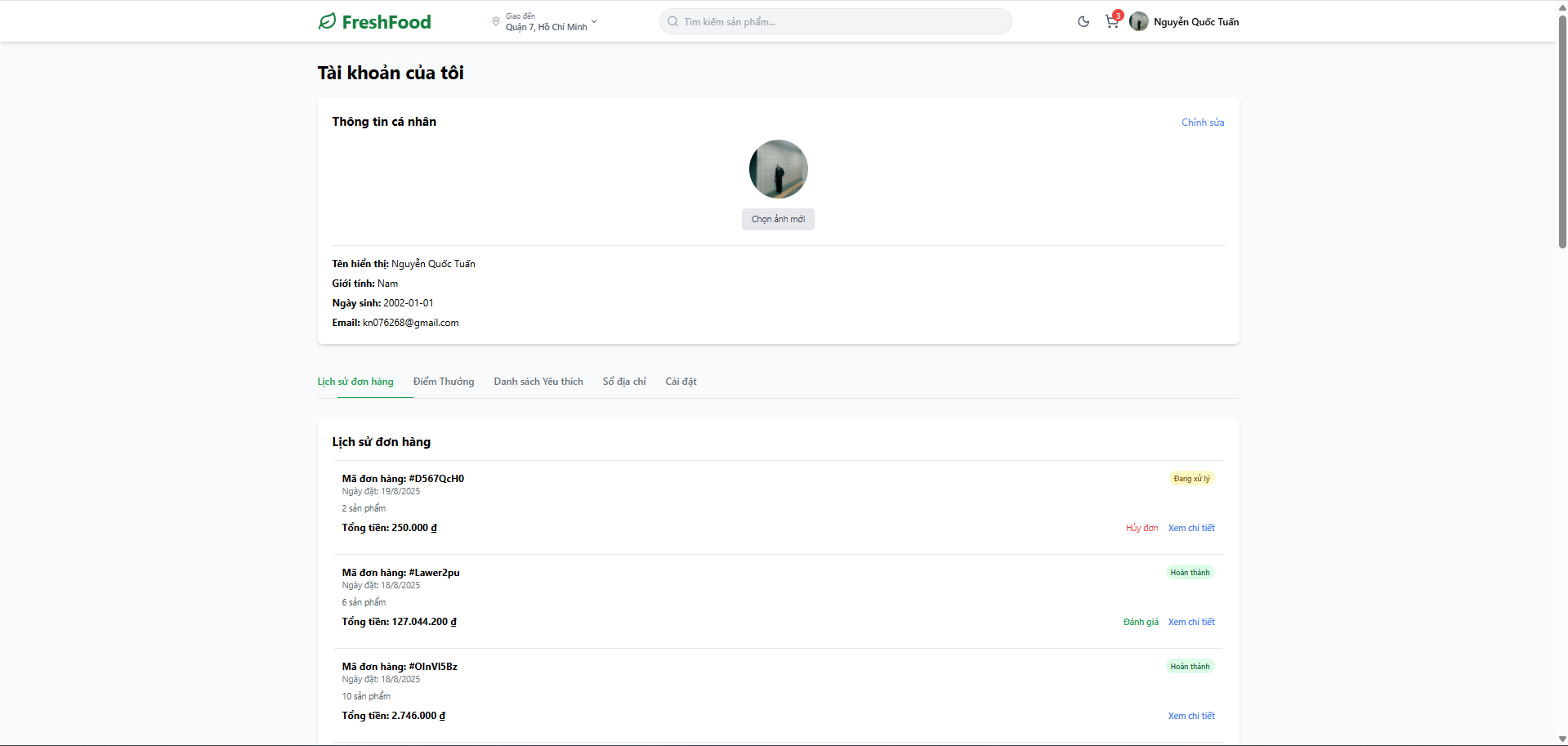
Hình 4.0.31 Trang giỏ hàng (CartPage)



Hình 4.0.32 Trang thanh toán (CheckoutPage)

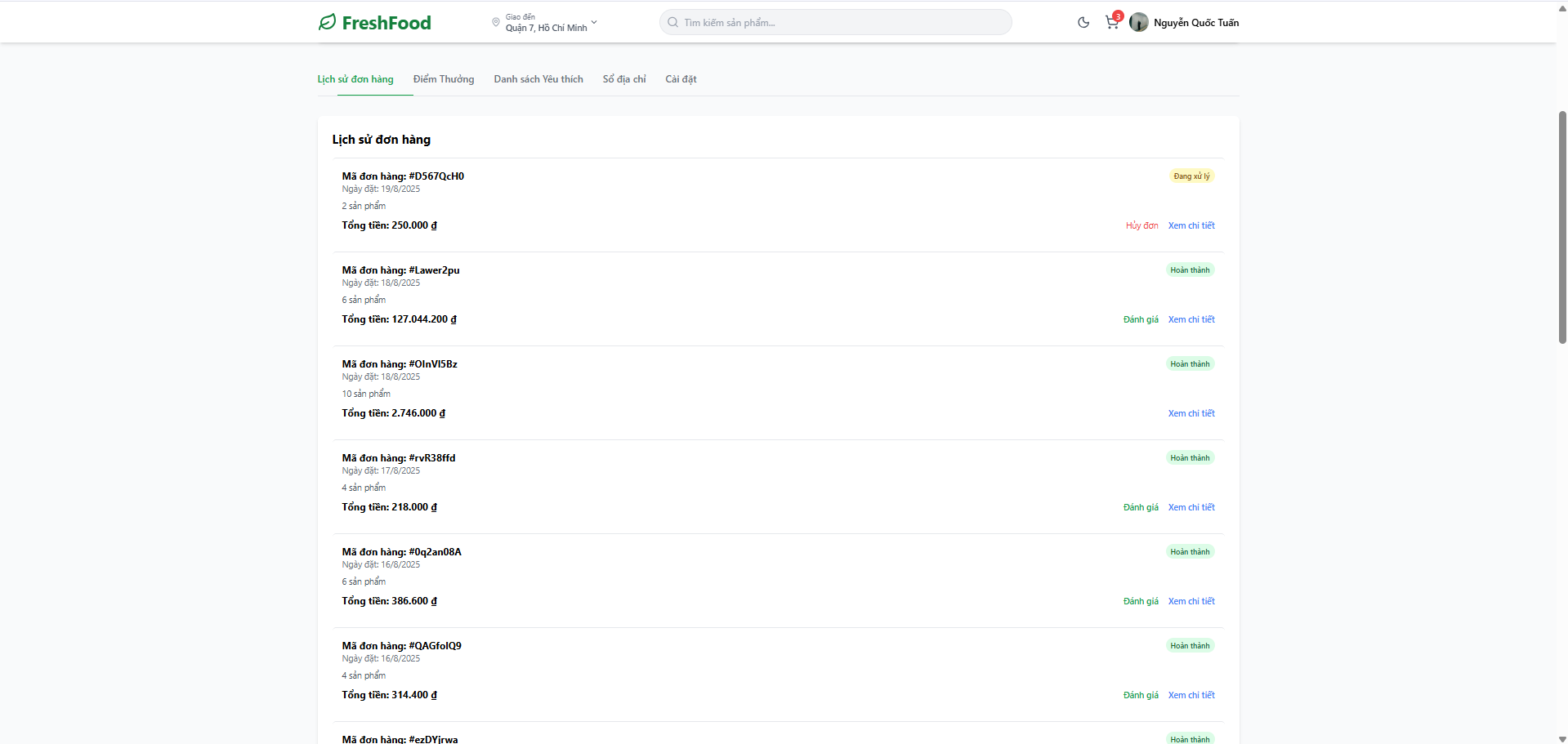
* **Chức năng:** Cung cấp quy trình mua hàng hoàn chỉnh, từ việc quản lý giỏ hàng đến khi hoàn tất đơn hàng.
* **Phân tích Code:**
  + **Đồng bộ Giỏ hàng:** Giỏ hàng được lưu trữ trong subcollection cart của người dùng trên Firestore. AppContext sử dụng onSnapshot để lắng nghe sự thay đổi của giỏ hàng theo thời gian thực, đảm bảo dữ liệu luôn được đồng bộ trên mọi thiết bị mà người dùng đăng nhập.
  + **Chọn món Thanh toán:** CartPage được trang bị checkbox cho mỗi sản phẩm, cho phép người dùng chọn những món hàng họ muốn thanh toán. State selectedItems được dùng để theo dõi các lựa chọn này. Khi nhấn "Mua hàng", chỉ những sản phẩm được chọn mới được gửi đến trang thanh toán.
  + **Mua Ngay:** Chức năng này cho phép người dùng bỏ qua giỏ hàng. Dữ liệu của sản phẩm "Mua Ngay" được truyền trực tiếp đến CheckoutPage thông qua page.data.
  + **Thanh toán:** CheckoutPage là một component phức tạp, xử lý nhiều logic:
    - **Xác định sản phẩm:** Tự động xác định nguồn sản phẩm cần thanh toán (từ giỏ hàng đã lọc, từ "Mua Ngay", hoặc toàn bộ giỏ hàng).
    - **Địa chỉ & Vận chuyển:** Lấy và cho phép người dùng chọn địa chỉ giao hàng đã lưu.
    - **Khuyến mãi:** Lấy danh sách các mã khuyến mãi hợp lệ, cho phép áp dụng và tự động tính toán lại tổng tiền.
    - **Tích hợp Thanh toán Online:** Hệ thống đã được nâng cấp để tích hợp cổng thanh toán Stripe. CheckoutPage giờ đây chứa component CardElement của Stripe để người dùng nhập thông tin thẻ một cách an toàn. Khi người dùng chọn thanh toán bằng thẻ, hàm handlePlaceOrder sẽ gọi đến backend (Vercel Function) để tạo phiên giao dịch, sau đó dùng clientSecret nhận về để xác nhận thanh toán trước khi lưu đơn hàng vào Firestore. Chức năng thanh toán qua mã QR đã được lược bỏ để tập trung vào luồng thanh toán thẻ và COD.
    - **Giao dịch An toàn:** Toàn bộ quá trình đặt hàng (kiểm tra tồn kho, tạo đơn hàng, cập nhật tồn kho) được thực hiện bên trong một runTransaction của Firestore. Điều này đảm bảo tính toàn vẹn dữ liệu: nếu một bước nào đó thất bại (ví dụ: hết hàng), toàn bộ giao dịch sẽ bị hủy bỏ.
    - **Xử lý sau Đặt hàng:** Sau khi thành công, hàm handlePlaceOrder sẽ gọi hàm removeItemsFromCart để chỉ xóa những sản phẩm đã mua khỏi giỏ hàng trên Firestore.

### 4.5.3. Tài khoản Người dùng (ProfilePage)

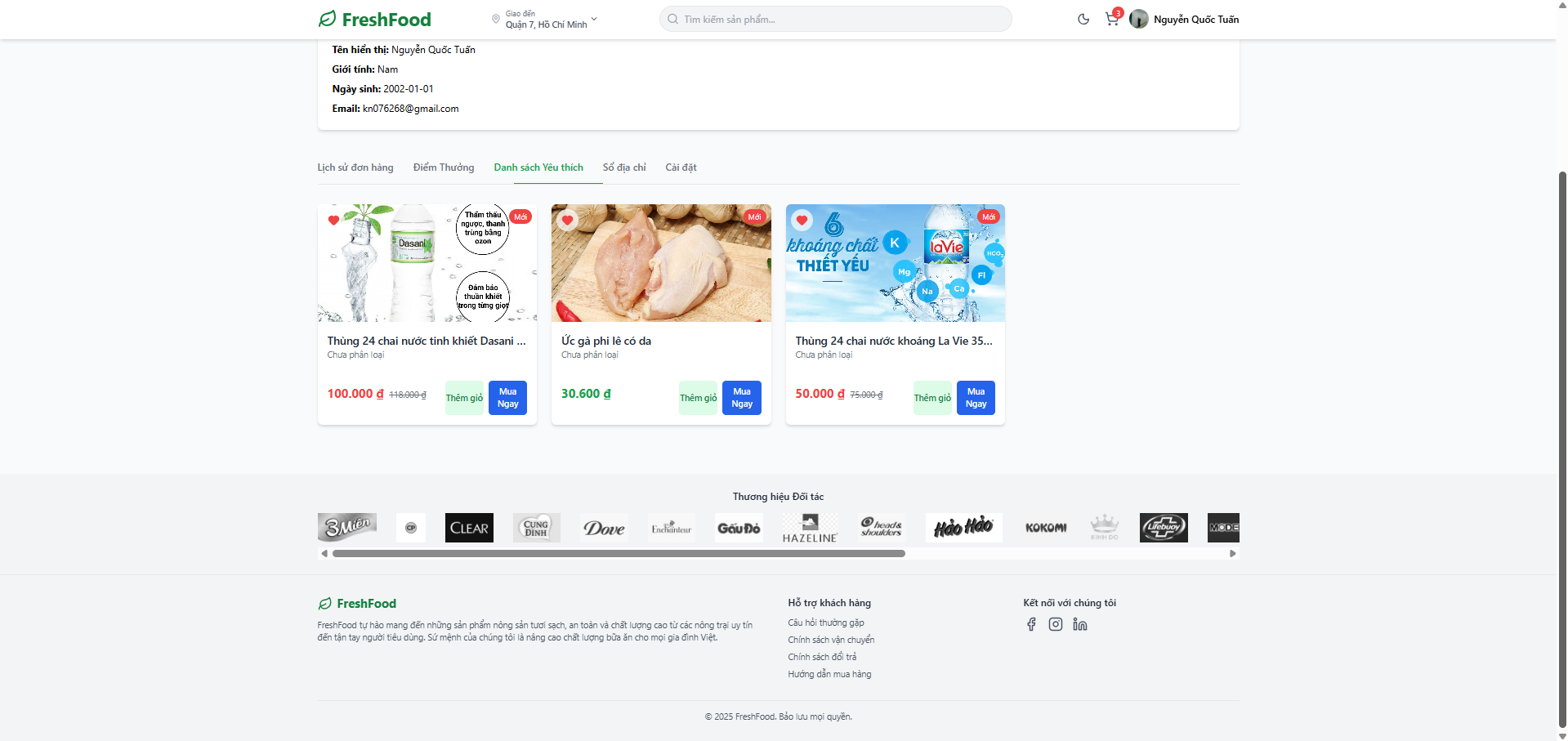


Hình 4.0.33 Trang tài khoản Người dùng (ProfilePage)

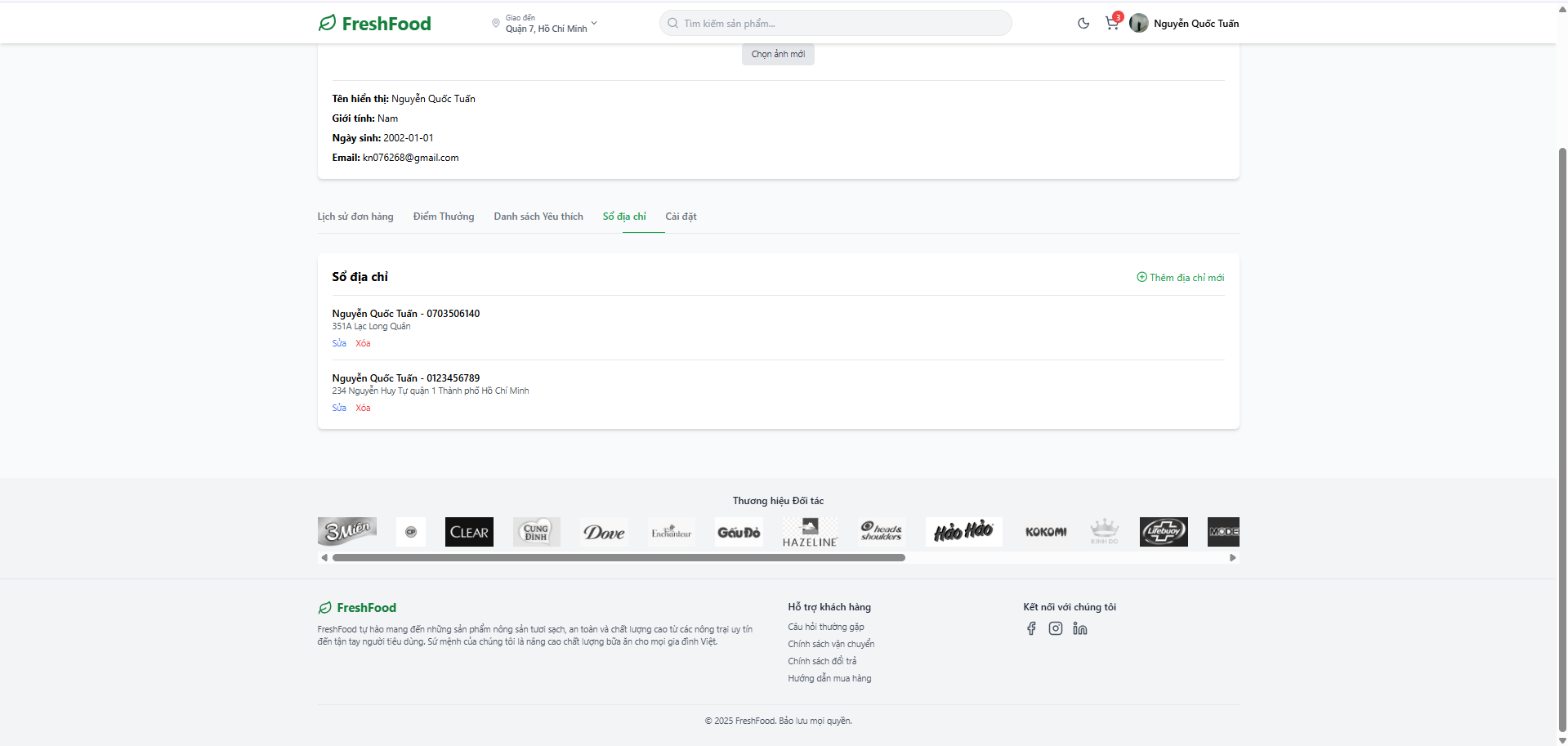
* **Chức năng:** Một khu vực cá nhân hóa nơi người dùng có thể quản lý thông tin, xem lịch sử và tương tác với các tính năng khác.
* **Phân tích Code:**
  + **Giao diện Tab:** Sử dụng state activeTab để chuyển đổi giữa các mục: Lịch sử đơn hàng, Danh sách Yêu thích, Sổ địa chỉ, Điểm thưởng và Cài đặt.



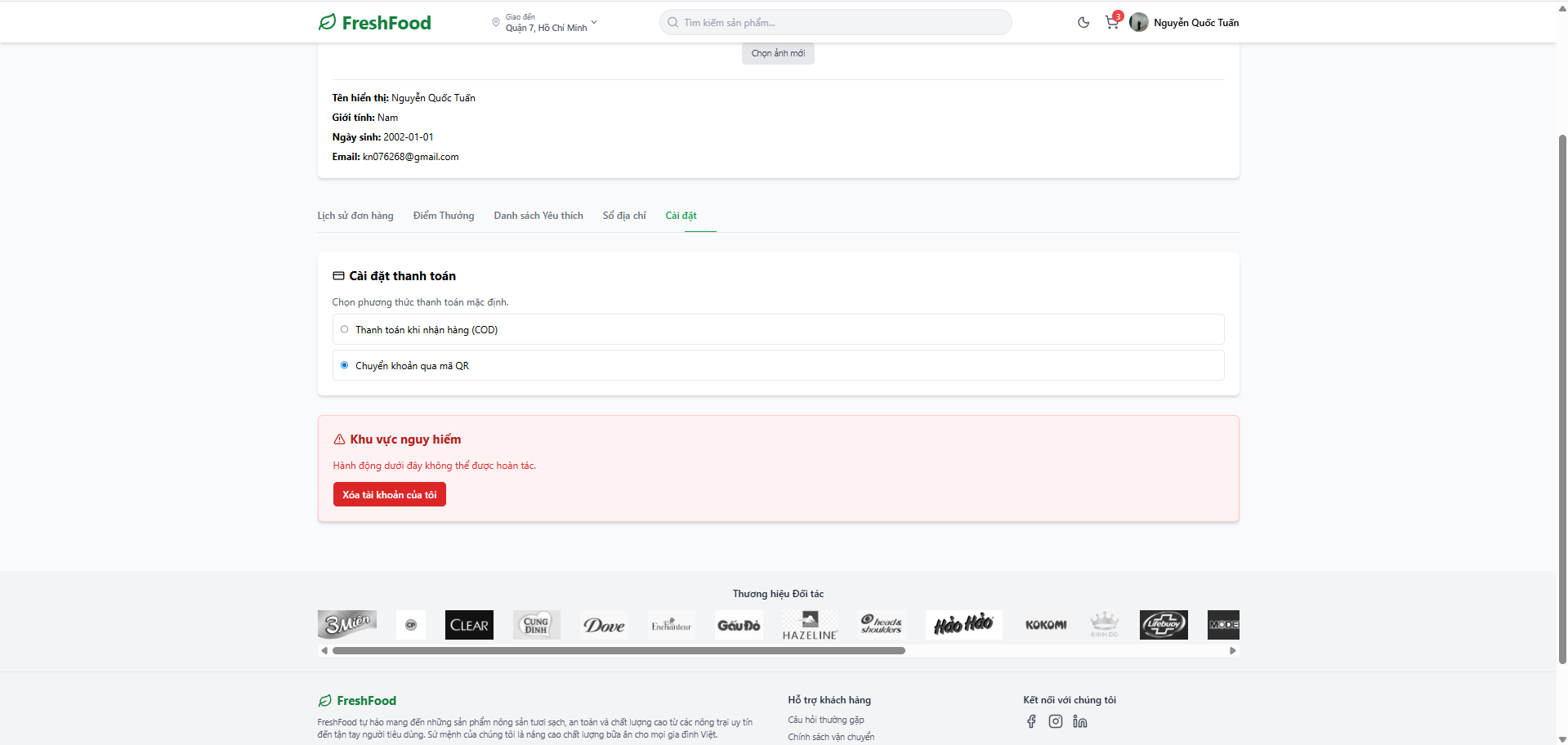
Hình 4.0.34 Trang lịch sử đơn hàng



Hình 4.0.35 Trang danh sách yêu thích

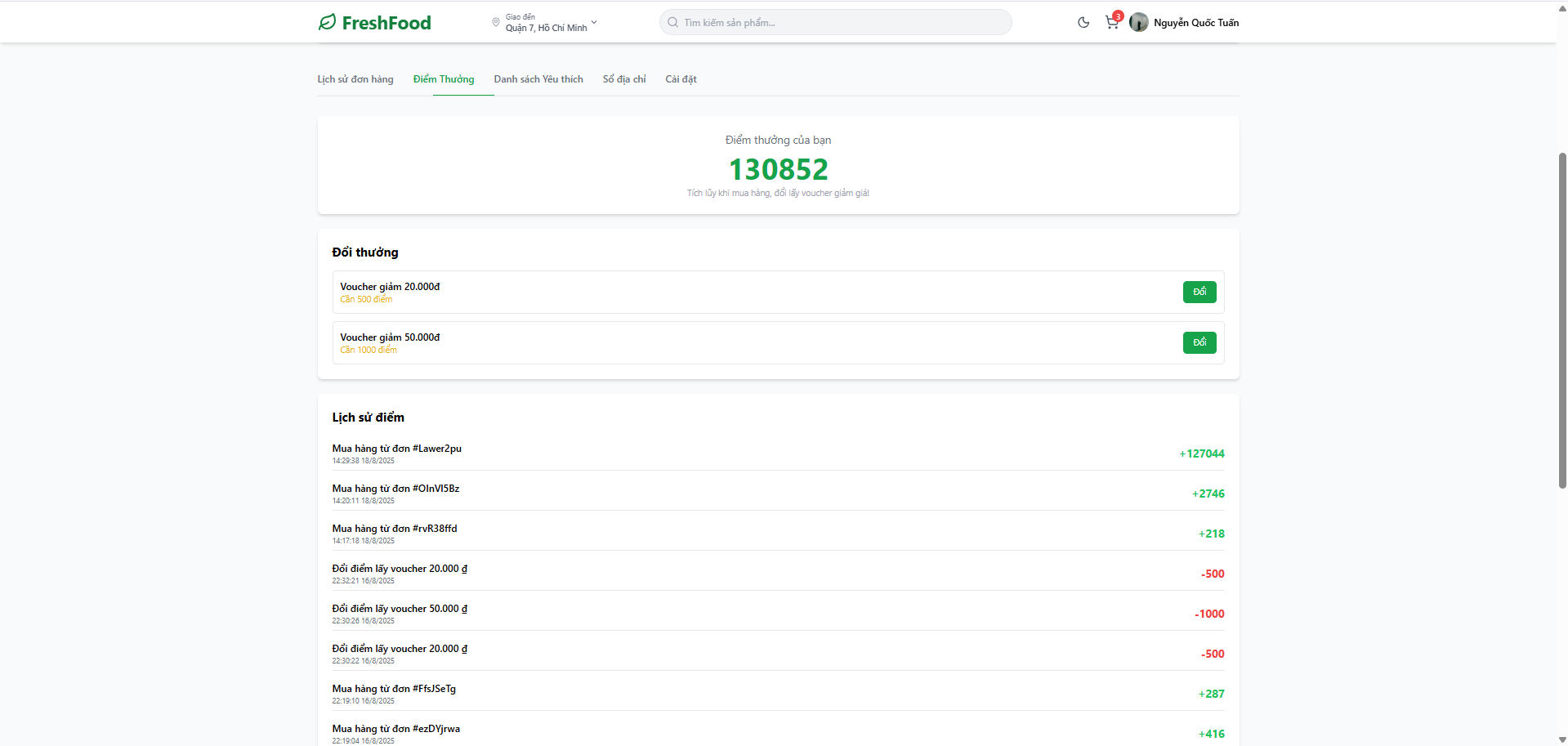


Hình 4.0.36 Trang danh sách địa chỉ



Hình 4.0.37 Trang cài đặt

* + **Hệ thống Điểm thưởng:** Tab "Điểm thưởng" hiển thị điểm hiện tại, lịch sử cộng/trừ điểm (lấy từ subcollection pointsHistory), và cho phép người dùng đổi điểm lấy voucher. Logic đổi điểm (handleRedeemPoints) được xử lý an toàn ở phía client, tạo ra một mã giảm giá mới trong collection promotions.



Hình 4.0.38 Trang đổi điểm thưởng

* + **Quản lý Thông tin:** Cho phép người dùng cập nhật tên hiển thị, ảnh đại diện, giới tính, ngày sinh và địa chỉ giao hàng.

### 4.5.4 Chi tiết sản phẩm (ProductDetailPage)

## 4.6. Module Hệ thống Quản trị (Admin)

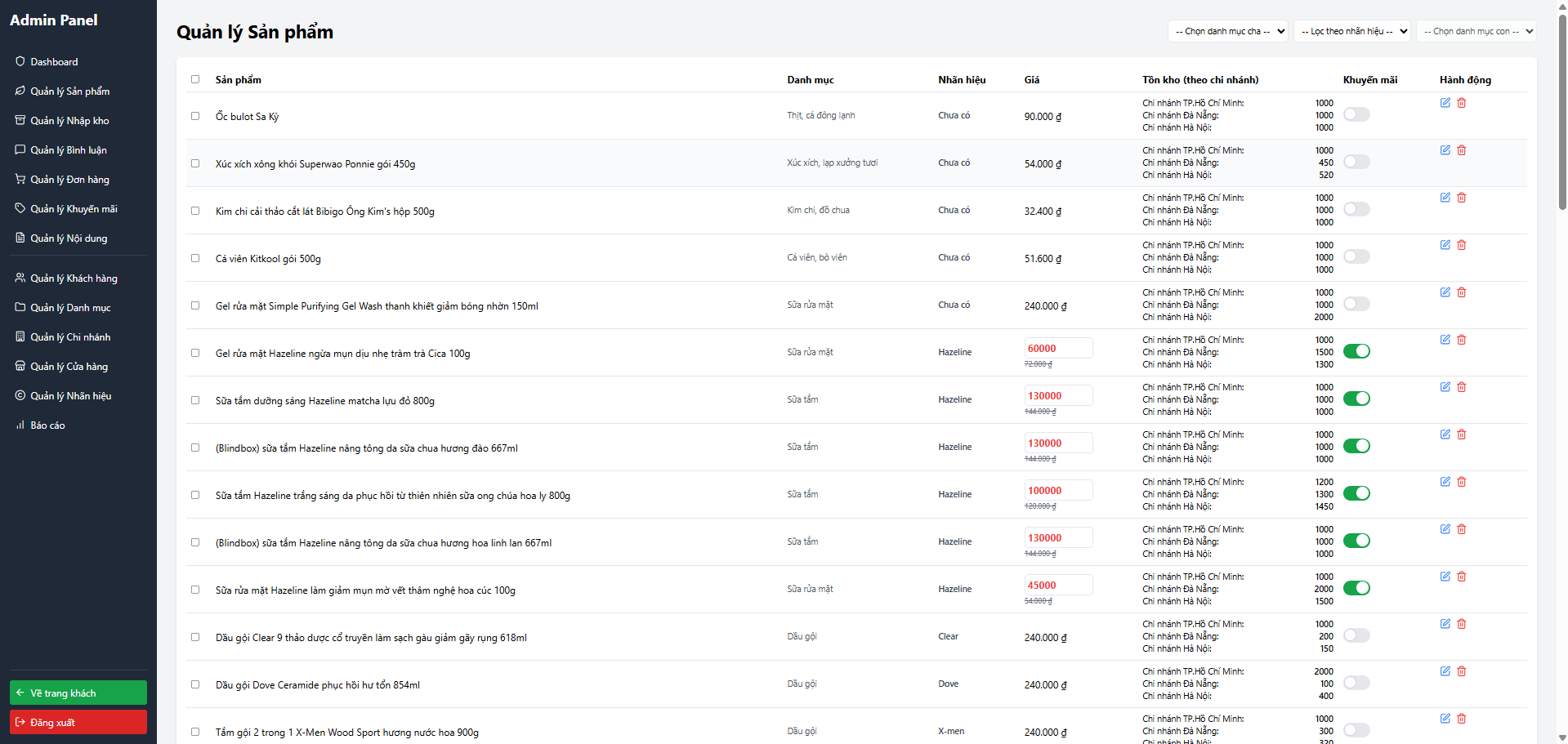
Hệ thống quản trị được phân quyền rõ ràng, cung cấp các công cụ mạnh mẽ để vận hành website.

### 4.6.1. Phân quyền và Layout

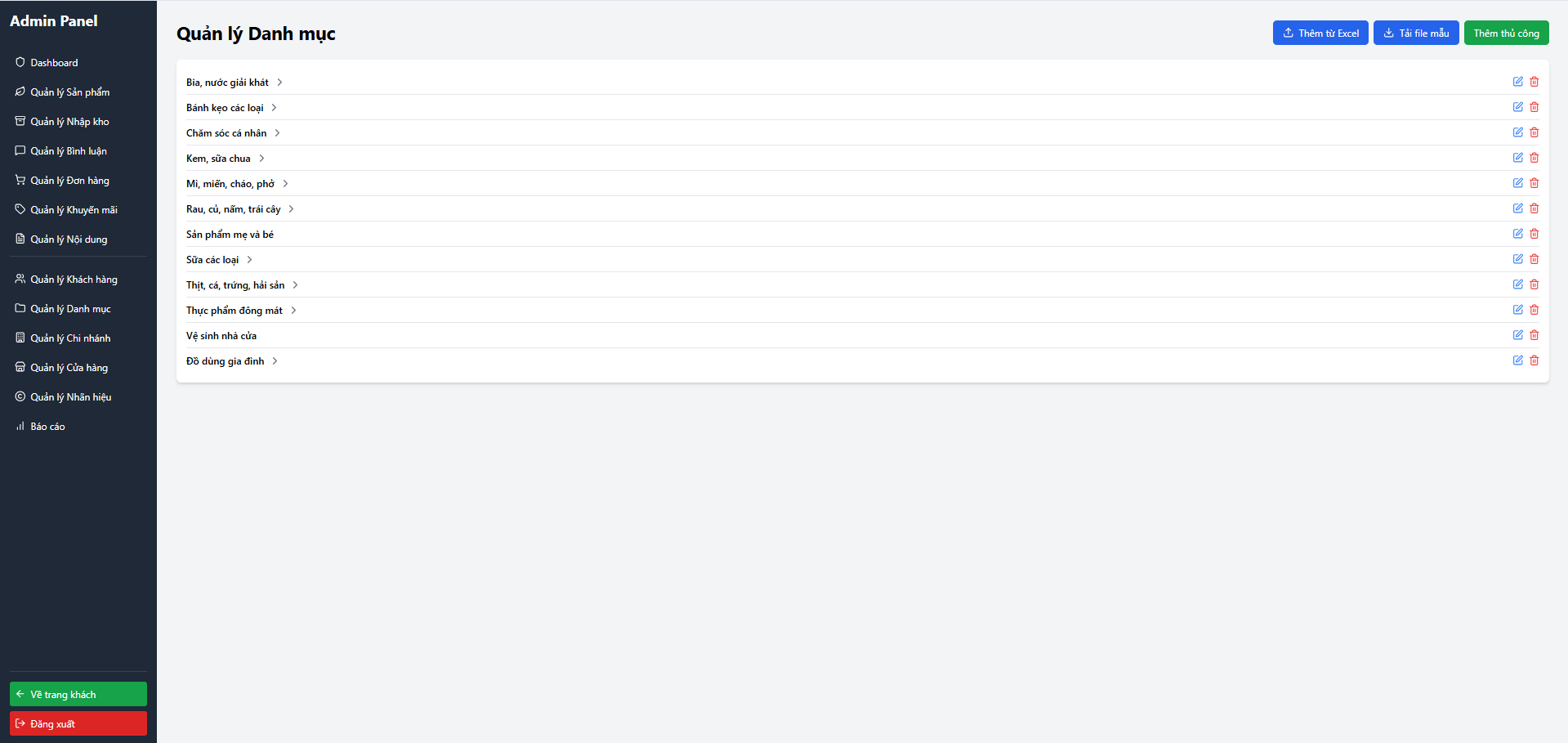
* **Phân quyền:** Dựa trên trường role trong users, hàm renderPage sẽ điều hướng người dùng đến giao diện phù hợp:
  + customer: Giao diện khách hàng.
  + staff, admin: Giao diện quản trị cấp cao.
* **Layout:** AdminLayout và VendorLayout cung cấp khung giao diện (sidebar, main content) nhất quán cho các trang quản trị.

### 4.6.2. Chức năng của Admin chính

* **Quản lý Sản phẩm (AdminProductsPage) và Danh mục (AdminProductsPage):**
  + Hiển thị danh sách tất cả sản phẩm với thông tin tồn kho chi tiết của từng chi nhánh.
  + Cung cấp bộ lọc 2 cấp (danh mục cha -> con) và lọc theo nhãn hiệu.
  + **Chức năng hàng loạt:** Cho phép chọn nhiều sản phẩm cùng lúc để gán nhãn hiệu, giúp tiết kiệm thời gian.

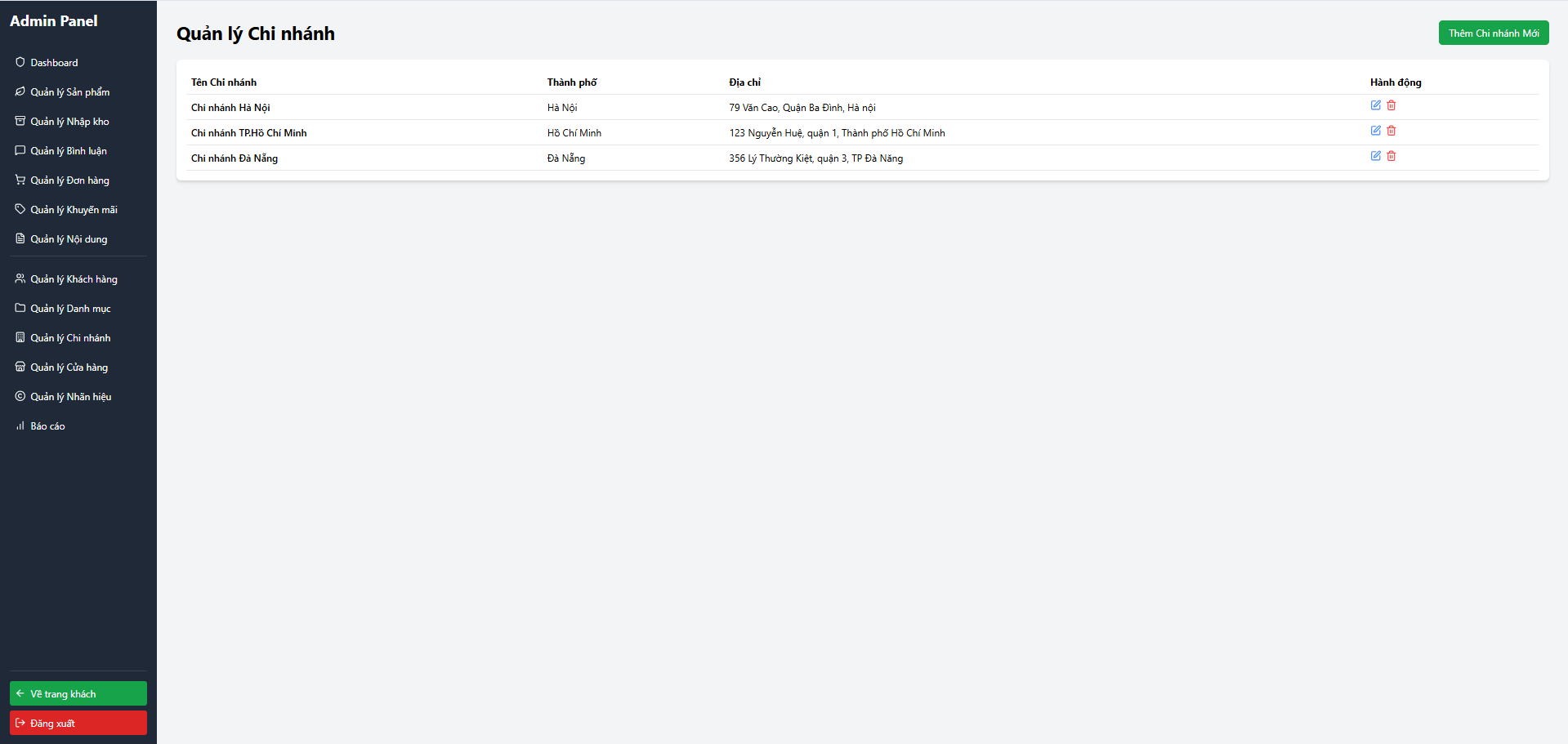


Hình 4.0.39 Trang quản lý sản phẩm

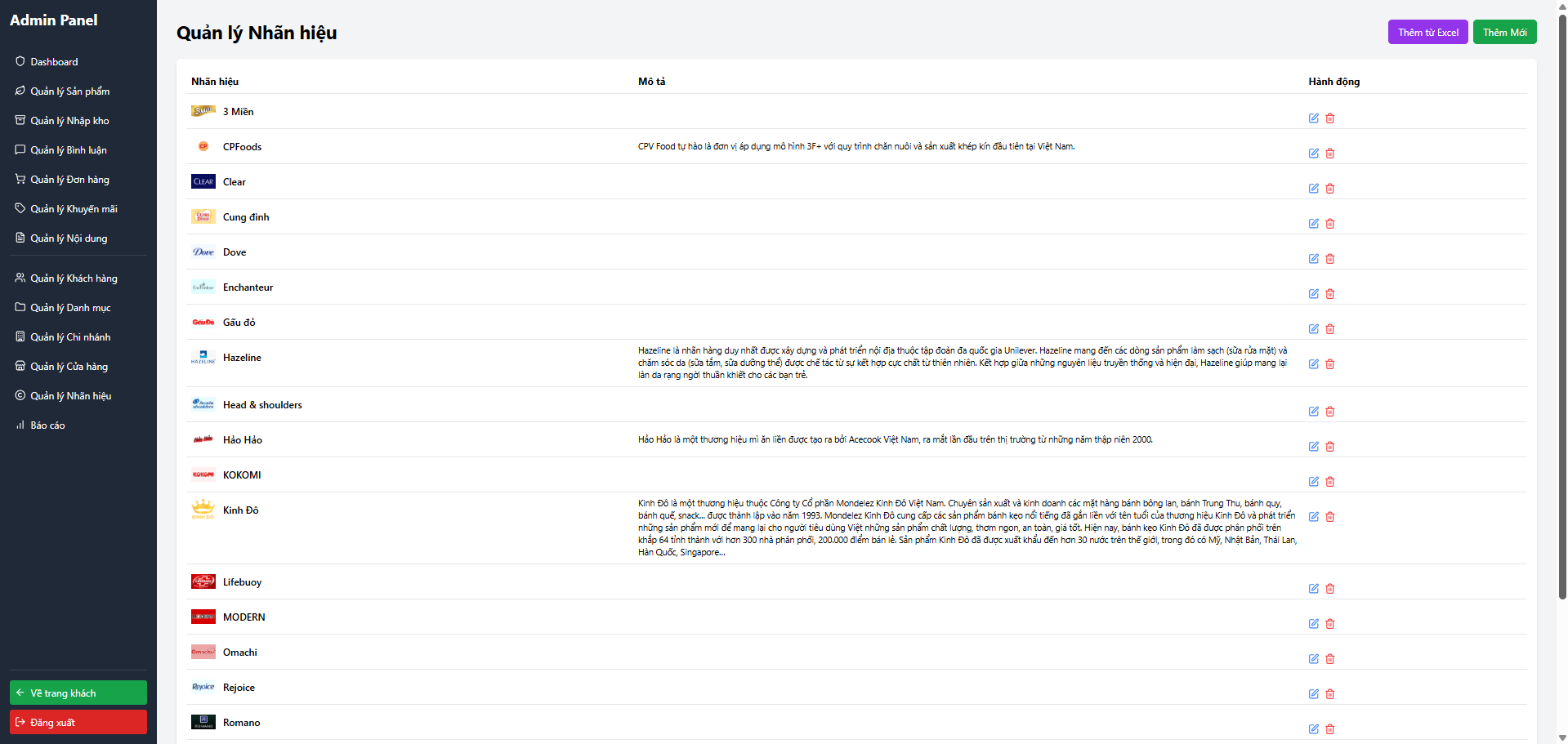


Hình 4.0.40 Trang quản lý danh mục

* **Quản lý Chi nhánh (AdminBranchesPage) và Nhãn hiệu (AdminBrandsPage):** Cung cấp các công cụ CRUD để quản lý các thực thể nền tảng này.

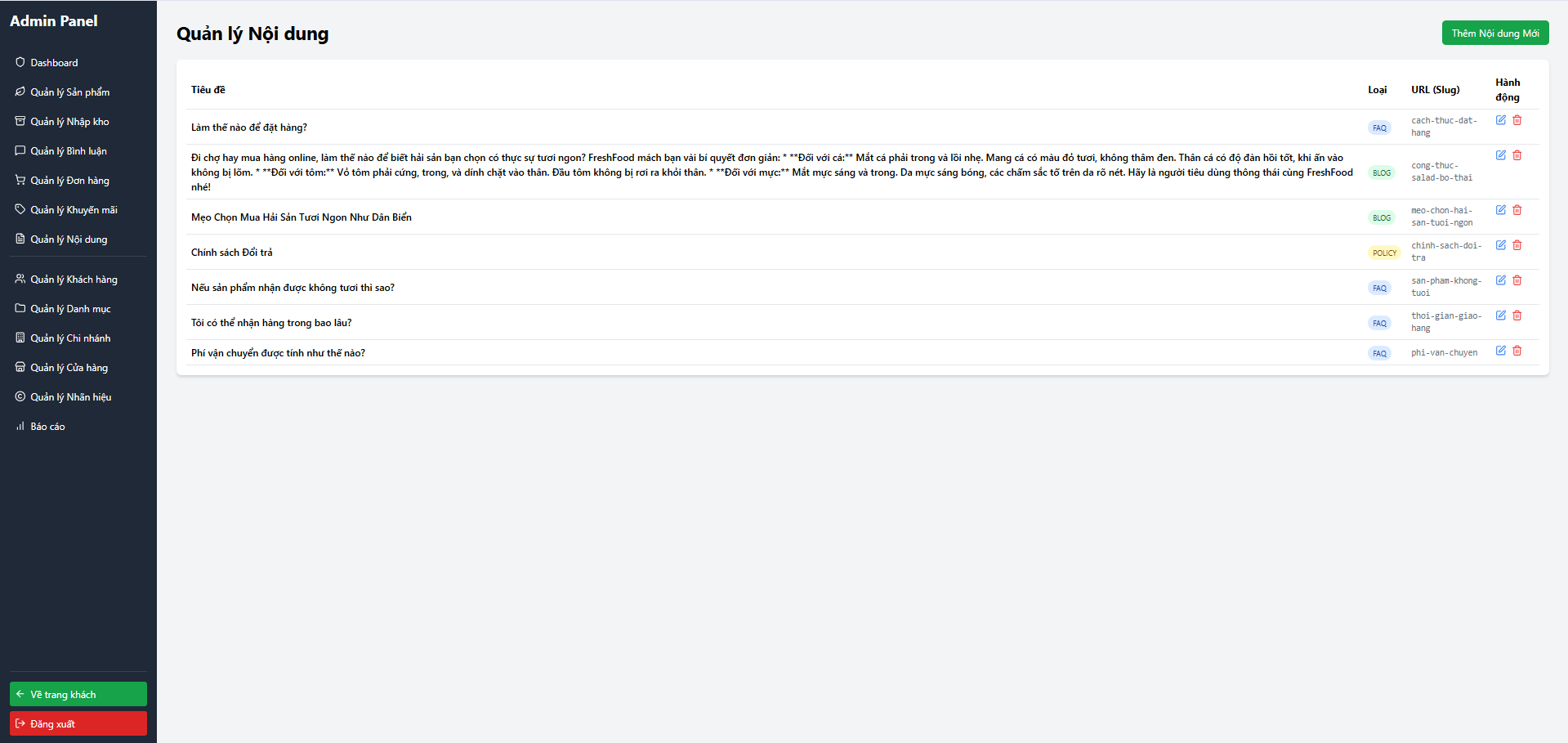


Hình 4.0.41 Trang quản lý chi nhánh



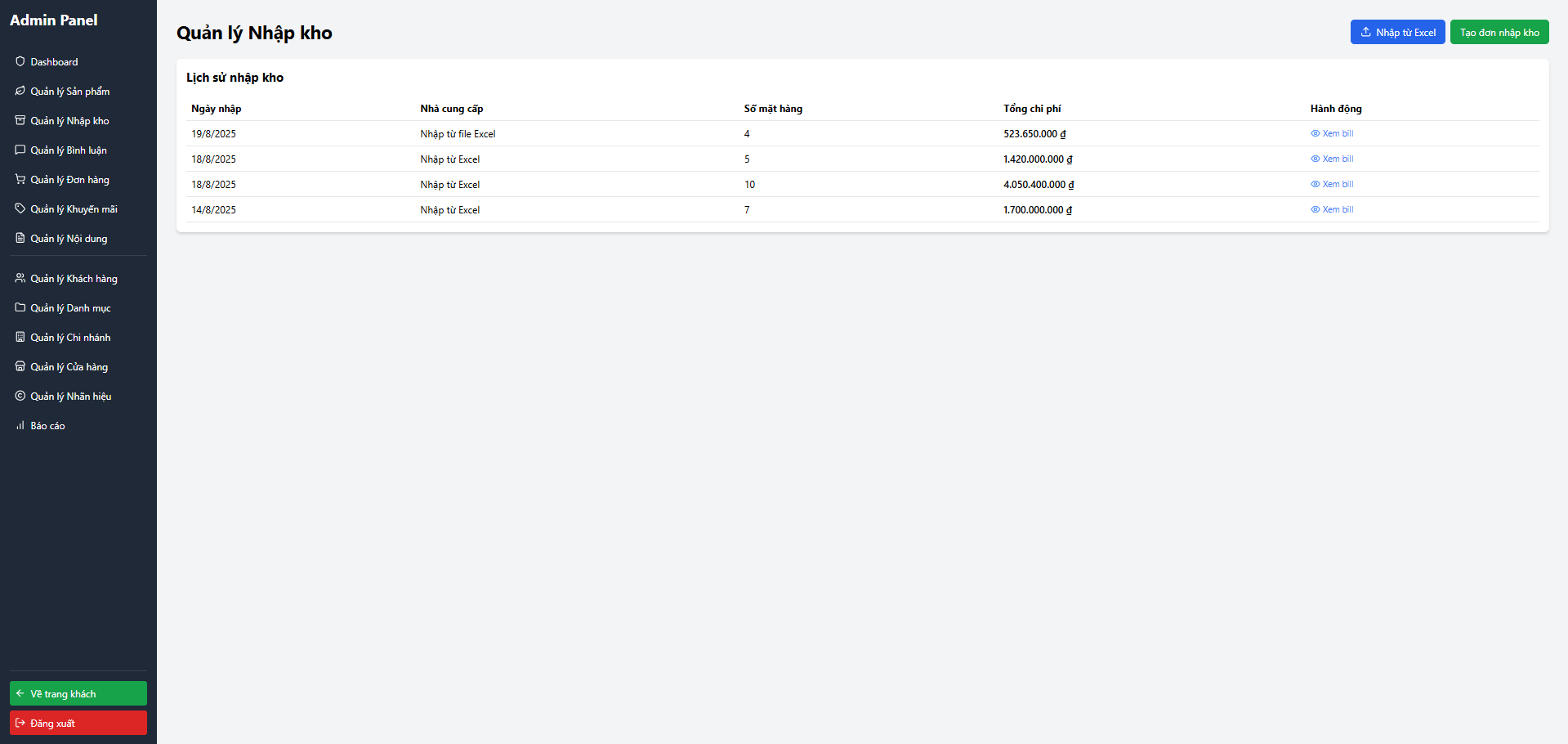
Hình 4.0.42 Trang quản lý nhãn hiệu

* **Quản lý Nội dung (AdminContentPage):** Một hệ thống CMS mini cho phép Admin tự thêm/sửa/xóa nội dung cho các trang FAQ, chính sách mà không cần can thiệp vào code.



Hình 4.0.43 Trang quản lý nội dung

* **Nhập liệu hàng loạt:** Các chức năng nhập sản phẩm, danh mục, và nhập kho từ file Excel sử dụng thư viện xlsx và ExcelJS, giúp tự động hóa các tác vụ tốn nhiều thời gian.



Hình 4.0.44 Trang quản lý nhập kho

## 4.7. Tích hợp các Module hỗ trợ và tối ưu hóa

### 4.7.1. Tối ưu hóa cho Công cụ Tìm kiếm (SEO) và Chia sẻ Mạng xã hội

**Vấn đề:** Ứng dụng trang đơn (SPA) ban đầu sử dụng cơ chế điều hướng dựa trên state (setPage), dẫn đến toàn bộ trang web chỉ có một URL duy nhất. Điều này gây ra các hạn chế nghiêm trọng:

* Người dùng không thể đánh dấu (bookmark) hay chia sẻ link đến một sản phẩm cụ thể.
* Các công cụ tìm kiếm như Google không thể lập chỉ mục (index) các trang con, làm giảm khả năng SEO.
* Khi chia sẻ link lên mạng xã hội (Facebook, Zalo), thông tin hiển thị (tiêu đề, ảnh, mô tả) không chính xác.

**Giải pháp:**

* **URL Thân thiện (URL Friendly):** Nâng cấp hệ thống điều hướng sang thư viện react-router-dom, là tiêu chuẩn cho các ứng dụng React hiện đại. Điều này giúp mỗi trang (sản phẩm, danh mục, giỏ hàng...) có một URL riêng biệt, có ý nghĩa (ví dụ:

/product/ten-san-pham), thân thiện với cả người dùng và công cụ tìm kiếm.

* **Meta Tags động:** Tích hợp thư viện react-helmet-async để quản lý các thẻ trong <head> của trang web. Giải pháp này cho phép mỗi trang có thể tự cập nhật các thẻ meta quan trọng như <title>, <meta name="description"> và các thẻ Open Graph (og:title, og:image...) một cách linh động dựa trên nội dung của trang đó.
* **Component <SEO>:** Xây dựng một component <SEO> có thể tái sử dụng để dễ dàng quản lý và áp dụng các thẻ meta cho mọi trang trong ứng dụng, đảm bảo tính nhất quán và dễ bảo trì.

### 4.7.2. Tích hợp Plugin Chat trực tuyến

* **Vấn đề:** Thiếu kênh tương tác và hỗ trợ khách hàng theo thời gian thực, làm giảm khả năng tư vấn và giải đáp thắc mắc ngay lập tức cho người mua.
* **Giải pháp:** Tích hợp dịch vụ chat trực tuyến miễn phí và chuyên nghiệp **Tawk.to**.
* **Phân tích Code:** Việc tích hợp được thực hiện một cách đơn giản và hiệu quả bằng cách nhúng đoạn mã (script) do Tawk.to cung cấp trực tiếp vào file public/index.html, ngay trước thẻ đóng </body>. Giải pháp này cho phép hiển thị một bong bóng chat trên toàn bộ trang web mà không cần can thiệp sâu vào mã nguồn React, giúp admin có thể trò chuyện và hỗ trợ khách hàng ngay lập tức thông qua dashboard của Tawk.to.

### 4.7.3. Xây dựng chức năng Email Marketing

* **Vấn đề:** Hệ thống ban đầu chưa có cơ chế thu thập email của khách hàng tiềm năng và không có công cụ để gửi các bản tin khuyến mãi, chăm sóc khách hàng một cách chuyên nghiệp.
* **Giải pháp:** Tích hợp toàn diện với dịch vụ email marketing chuyên nghiệp **Brevo (trước đây là Sendinblue)**. Quá trình này bao gồm hai phần chính:
  1. **Thu thập Email:** Xây dựng một component React (NewsletterForm.js) cho phép người dùng nhập email để đăng ký nhận tin. Component này được đặt ở chân trang (Footer) và có thể được sử dụng ở một trang đăng ký riêng biệt để tối ưu hóa việc thu thập.
  2. **Lưu trữ Contact an toàn:** Một **Vercel Serverless Function** (api/subscribe-newsletter.js) được tạo ra để làm trung gian. Function này nhận email từ frontend, sau đó sử dụng API Key của Brevo (được lưu an toàn dưới dạng biến môi trường trên Vercel) để thêm email của người dùng vào danh sách liên hệ (Contact List) đã định sẵn trên Brevo. Cách làm này đảm bảo mã API Key bí mật không bị lộ ra phía client, tuân thủ các tiêu chuẩn bảo mật.
  3. **Gửi Email hàng loạt:** Việc soạn thảo và gửi email marketing (chiến dịch) được thực hiện hoàn toàn trên nền tảng dashboard của Brevo. Admin có thể sử dụng công cụ kéo-thả trực quan để thiết kế email, chọn danh sách người nhận đã thu thập được, và lên lịch gửi mà không cần can thiệp thêm vào code. Giải pháp này tận dụng được các tính năng mạnh mẽ của Brevo như quản lý hủy đăng ký, thống kê hiệu quả chiến dịch (tỷ lệ mở, tỷ lệ nhấp chuột), và đảm bảo khả năng gửi email vào hộp thư đến cao.

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN HỆ THỐNG

## 5.1. Kết luận

Qua quá trình thực hiện đồ án, hệ thống website thương mại điện tử FreshFood đã được xây dựng thành công, đáp ứng đầy đủ và vượt qua các mục tiêu ban đầu đề ra.

### 5.1.1. Những kết quả đạt được

* **Hoàn thiện một nền tảng TMĐT đầy đủ chức năng:** Hệ thống đã xây dựng thành công một website B2C hoàn chỉnh với trải nghiệm người dùng mượt mà (phía client) và một hệ thống quản trị mạnh mẽ (phía admin). Các module cốt lõi như sản phẩm, giỏ hàng, thanh toán, đơn hàng, và quản lý người dùng đều hoạt động ổn định và logic.
* **Triển khai thành công các mô hình kinh doanh phức tạp:**
  + **Mô hình đa chi nhánh:** Hệ thống đã quản lý thành công tồn kho và đơn hàng theo từng chi nhánh riêng biệt, một yếu tố quan trọng cho việc vận hành thực tế.
  + **Mô hình đa nhà cung cấp:** Đã xây dựng được nền tảng vững chắc cho một sàn TMĐT, với hệ thống phân quyền rõ ràng cho phép các nhà cung cấp (vendor) tự quản lý sản phẩm và đơn hàng của riêng họ.
* **Áp dụng hiệu quả công nghệ hiện đại:** Việc sử dụng ReactJS và Google Firebase đã chứng tỏ được tính ưu việt trong việc rút ngắn thời gian phát triển, cung cấp hiệu năng cao và khả năng mở rộng linh hoạt mà không đòi hỏi chi phí hạ tầng lớn.
* **Xây dựng nhiều tính năng nâng cao:** Hệ thống không chỉ dừng lại ở các chức năng cơ bản mà còn triển khai các tính năng nâng cao nhằm tăng tương tác và giữ chân khách hàng như: hệ thống điểm thưởng, danh sách yêu thích, đánh giá sản phẩm, và quản lý nội dung động (CMS).
* **Tích hợp thành công Cổng thanh toán:** Hệ thống đã tích hợp thành công cổng thanh toán Stripe, cho phép xử lý giao dịch thẻ quốc tế một cách an toàn thông qua Vercel Serverless Functions, hoàn thiện quy trình mua hàng E-commerce tiêu chuẩn.
* **Email Marketing:** Xây dựng luồng thu thập email (qua form đăng ký) và gửi email hàng loạt (qua trang quản trị) một cách hoàn chỉnh, sử dụng dịch vụ chuyên nghiệp Brevo và Vercel Serverless Functions để đảm bảo tính bảo mật và hiệu quả.

### 5.1.2. Hạn chế của hệ thống

Bên cạnh những kết quả đạt được, hệ thống vẫn còn một số hạn chế nhất định do giới hạn về thời gian và phạm vi của đồ án:

* **Điều hướng phía Client:** Việc sử dụng cơ chế điều hướng dựa trên state (setPage) tuy đáp ứng được yêu cầu nhưng thiếu các tính năng quan trọng như URL duy nhất cho mỗi trang, không hỗ trợ nút Back/Forward của trình duyệt, và không thân thiện với SEO.
* **Tối ưu hóa Hiệu năng:** Với các danh sách lớn (sản phẩm, đơn hàng), hệ thống hiện đang tải toàn bộ dữ liệu về client rồi mới phân trang. Điều này có thể gây chậm khi dữ liệu tăng lên.
* **Thiếu các Tích hợp Nâng cao:** Hệ thống mới chỉ tích hợp một cổng thanh toán qua thẻ (Stripe) và chưa có các tùy chọn thanh toán qua ví điện tử phổ biến tại Việt Nam như MoMo, ZaloPay. Ngoài ra, hệ thống chưa tích hợp các dịch vụ giao hàng của bên thứ ba và chưa có các thuật toán gợi ý sản phẩm thông minh.
* **Thiếu Tự động hóa Marketing:** Chức năng gửi email marketing hiện tại đang được kích hoạt thủ công bởi admin. Hệ thống chưa có các kịch bản tự động như gửi email chào mừng khi người dùng mới đăng ký, hay email nhắc nhở giỏ hàng bị bỏ quên.

## 5.2. Hướng phát triển hệ thống

Dựa trên nền tảng vững chắc đã xây dựng, hệ thống FreshFood có rất nhiều tiềm năng để phát triển và hoàn thiện trong tương lai. Dưới đây là một số hướng đi chính:

* **Nâng cấp Kỹ thuật (Ưu tiên hàng đầu):**
  + **Tích hợp React Router DOM:** Thay thế hệ thống setPage hiện tại bằng thư viện định tuyến tiêu chuẩn này để có URL thật, cải thiện trải nghiệm người dùng và tối ưu hóa cho SEO.
  + **Triển khai Phân trang phía Server:** Cải thiện hiệu năng bằng cách sửa lại các câu truy vấn để chỉ tải về lượng dữ liệu cần thiết cho mỗi trang, sử dụng các hàm limit() và startAfter() của Firestore.
  + **Tách file (Code Refactoring):** Tách file App.js khổng lồ hiện tại thành một cấu trúc thư mục rõ ràng theo từng component, page, và context để dễ dàng bảo trì và phát triển theo nhóm.
* **Hoàn thiện Trải nghiệm Mua sắm:**
  + **Đồng bộ Giỏ hàng:** Chuyển giỏ hàng từ localStorage sang lưu trữ trên Firestore để đồng bộ giữa các thiết bị của người dùng.
  + **Tích hợp Cổng thanh toán:** Tích hợp với các ví điện tử phổ biến như MoMo, ZaloPay để tăng sự tiện lợi cho khách hàng.
  + **Tìm kiếm Nâng cao:** Xây dựng chức năng tìm kiếm thông minh hơn với gợi ý tự động (autocomplete) và khả năng chịu lỗi chính tả.
* **Mở rộng Mô hình Kinh doanh:**
  + **Hoàn thiện Dashboard cho Vendor:** Xây dựng các biểu đồ thống kê chi tiết hơn cho người bán, cung cấp các công cụ quản lý khuyến mãi riêng cho cửa hàng của họ.
  + **Xây dựng hệ thống Hoa hồng:** Tự động tính toán và theo dõi doanh thu, hoa hồng cho từng nhà cung cấp.
  + **Hệ thống Nhắn tin:** Cho phép khách hàng và người bán có thể trao đổi trực tiếp với nhau về sản phẩm.
* **Tăng cường Tương tác và Marketing:**
  + **Xây dựng Blog:** Tận dụng hệ thống CMS đã có để xây dựng một chuyên mục Blog/Tin tức hoàn chỉnh với các bài viết về công thức, mẹo vặt, giúp thu hút người dùng tự nhiên từ các công cụ tìm kiếm.
  + **Hệ thống Thông báo:** Xây dựng một trung tâm thông báo (biểu tượng chuông) để gửi các cập nhật quan trọng (trạng thái đơn hàng, khuyến mãi mới) đến người dùng.
  + **Xây dựng hệ thống Marketing Automation:** Nâng cấp chức năng email marketing bằng cách tạo ra các luồng tự động (automation workflows) trên Brevo, được kích hoạt bởi các sự kiện trong ứng dụng.

Với những định hướng phát triển này, website FreshFood hoàn toàn có khả năng trở thành một nền tảng thương mại điện tử mạnh mẽ, cạnh tranh và đáp ứng tốt nhu cầu ngày càng cao của thị trường.